A RAND NOTE

AURA USER'S MANUAL: VOL. I,
PROGRAM FEATURES AND INTERACTIONS

Robert Shishko, Milton Kamins

June 1983

N-1987-MRAL

Prepared for

The Office of the Assistant Secretary of Defense/Manpower, Reserve Affairs and Logistics

Adapted for Ground Forces Applications from the TSAR User's Manual by D. E. Emerson

35th Year

Rand
SANTA MONICA, CA. 90406

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **JUN 1983**	2. REPORT TYPE	3. DATES COVERED **00-00-1983 to 00-00-1983**

4. TITLE AND SUBTITLE **AURA User's Manual: Vol. I, Program Features and Interactions**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Rand Corporation,1776 Main Street, PO Box 2138,Santa Monica,CA,90407-2138**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**152**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

The research described in this report was sponsored by the Office of the Assistant Secretary of Defense/Manpower, Reserve Affairs and Logistics under Contract No. MDA903-83-C-0047.

Published by The Rand Corporation

A RAND NOTE

AURA USER'S MANUAL: VOL. I,
PROGRAM FEATURES AND INTERACTIONS

Robert Shishko, Milton Kamins

June 1983

N-1987-MRAL

Prepared for

The Office of the Assistant Secretary of Defense/Manpower, Reserve Affairs and Logistics

Adapted for Ground Forces Applications from the TSAR User's Manual by D. E. Emerson

35th Year

Rand
SANTA MONICA, CA. 90406

PREFACE

This Note is one of five documents that collectively describe the AURA computer model, which can be used to assess the effect of resources on the mission-generation capabilities of combined arms units. AURA is an adaptation of the TSAR model developed by D. E. Emerson at The Rand Corporation under Project AIR FORCE sponsorship. This adaptation was carried out under a project for the Assistant Secretary of Defense (Manpower, Reserve Affairs and Logistics) entitled "Relating Resources to the Readiness and Sustainability of Army Units."

The Army Unit Readiness/Sustainability Assessor (AURA) model provides an analytic context within which a variety of support improvements may be tested. Alternative maintenance and supply doctrines, manpower policies, improved battle damage and recovery capabilities, and increased stock levels for parts and equipment, as well as concepts for improved theater-wide resource management, can be examined for their effect on mission generation.

The present Note provides a full description of the logic used in the AURA model, as well as an understanding of interrelationships among the many elements of the logic, for programmers interested in modifying and extending the existing program logic. Companion Rand documents relating to AURA and AURA-compatible models include:

- R-2769-MRAL, *Relating Resources to the Readiness and Sustainability of Combined Arms Units*, December 1981.

- N-1460-AF, *TSARINA: User's Guide to a Computer Model for Damage Assessment of Complex Airbase Targets*, July 1980.

- N-1988-MRAL, *AURA User's Manual: Vol. II, Data Input and Sample Problem*, May 1983.

- N-1999-MRAL, *AURA Applications: Division-Level Transportation and Selected Spares Issues*, forthcoming.

ABSTRACT

AURA (Army Unit Readiness/Sustainability Assessor) is a Monte Carlo discrete-event simulation model intended for analyzing the interrelationships among the resources associated with a set of combat units, and the capability of those units to generate combat missions in a dynamic, rapidly evolving wartime environment.

This volume is the first of two being prepared as a User's Manual for AURA. It provides a succinct but complete discussion of the capabilities of the model and the processing logic involved in the twelve major subsets of tasks. Discussion of the input requirements, procedures, and formats are to be found in Vol. II; these detailed discussions provide the only complete explanations for some of the numerous control options available with AURA and must be considered mandatory reading for any one planning to operate AURA.

ACKNOWLEDGMENTS

Our ability to adapt an existing large simulation model to represent Army combined arms operations depended heavily on the patient and cheerful cooperation of a substantial number of Army officers and enlisted personnel at many locations in the United States and Europe. We are indebted to a number of our Rand colleagues, particularly to Donald Emerson for allowing us to use the TSAR User's Manual as the basis for this Note, and for his patience in clarifying for us the subtler points in his original manuscript; to Robert Paulson for his data development assistance; and to Patricia Berger for implementing many of the program changes required. The timely completion of the documentation was ensured by the capable efforts of Barbara Urwin, Linda Freeman, and Mary Jane Digby.

GLOSSARY

ASL	Authorized Stockage List; DS level spares, typically for a division, part of which are located at FASTs and part at the DISCOM
ATE	Automatic Test Equipment; special test equipment used for repairing complex LRUs and SRUs
AURA	Army Unit Readiness/Sustainability Assessor
CONUS	Continental United States
COSCOM	Corps Support Command
DISCOM	Division Support Command
DS	Direct Support
FAST	Forward Area Support Team
FRAG	Fragmentary order that specifies mission requirements
GS	General Support
GSRF	General Support Repair Facility
HET	Heavy Equipment Transporter
LCOM	Logistics Composite Model
LRU	Line replaceable unit; a vehicle spare part
NMCS	Not mission capable because of lack of spare parts (deadlined)
NRTS	Not reparable this station
OST	Order and ship time
OWRMS	Other War Reserve Materiel Stocks
PLL	Prescribed Load List; organizational level spares distributed to battalions and companies
POL	Petroleum, oils and lubricants; often used as an abbreviation for fuel

PWRMS	Prepositioned War Reserve Materiel Stocks
RR	Maintenance that removes and replaces malfunctioning vehicle parts with serviceable components
RRR	Maintenance that removes, repairs, and replaces spare parts (actually, usually removes and replaces with a serviceable unit, and then repairs malfunctioning unit)
SAMSOM	Support Availability Multi-System Operations Model
SCL	Standard combat load that designates the mission dependent munitions to be loaded
SOC	Specific Operational Capability; a "notational" operation
SRU	Shop replaceable unit; a component of an LRU
TSAR	Theater Simulation of Airbase Resources
TSARINA	TSAR Inputs using AIDA
TTU	TSAR(AURA) Time Units; three minutes

CONTENTS

I. SUMMARY OF AURA CAPABILITIES

AURA simulates a system of interdependent maneuver, artillery, and aviation units organized into a brigade, division, or corps structure, and supported by shipments from CONUS and by intra-theater transportation, communication, and resource management systems. The simulation, by capturing the interdependencies among eleven classes of resources, will permit decisionmakers to examine the implications of a broad spectrum of possible improvements, in terms of their effect on mission-generation capabilities. The simulation also allows examination of the effects of damage inflicted by enemy attacks and by efforts to restore operations.

The classes of resources treated in AURA are (1) the weapon system or other unit pacing item, (2) crews, (3) support personnel, (4) support equipment, (5) spare parts, (6) weapon system shelters, (7) munitions, (8) accessories, (9) POL, (10) building and barrier materials, and (11) fixed facilities. Many different types within each class of resource may be distinguished. When included in the simulation, either the user will specify initial parts stocks or AURA will initialize the parts data according to standard algorithms for spares availability and will also initialize the stock location in the depot pipeline.

AURA is a Monte Carlo discrete-event simulation model that analyzes the interrelations among available resources and the capability of the units to generate weapon system missions in a dynamic, rapidly evolving wartime environment. On-vehicle maintenance tasks, parts and support equipment repair jobs, munitions assembly, and facilities repair tasks are simulated at each of several maneuver, artillery, and aviation units. A broad range of policy options that would increase initial resources, accelerate task completion, or improve theater resource utilization may be assessed using AURA. The user can also assess dynamic variations in key management policies.

AURA is readily adaptable to a broad range of initial conditions. When specific features are not needed for the examination of a particular issue, they simply need not be used. Thus, AURA permits a

user to represent a single unit, a set of independent units, or a set of interdependent units, without any adjustment or modification of the program. Similarly, the user may not wish to examine the effects of enemy air and artillery attacks on rear support areas, or may wish to ignore the possible constraints imposed by shortages of crews, maintenance personnel, support equipment, spare parts, munitions, accessories, and/or fuel. AURA adapts automatically to all such problem representations.

AURA provides users a means with which a rich variety of resource sets may be tested in a common context. Alternative maintenance and supply doctrines, manpower policies, and increased stock levels for parts and equipment, as well as several concepts for theater-wide resource management, can be assessed with AURA by comparing how such changes affect the ability of combined arms units to generate missions.

An important objective in the original design formulation of TSAR, AURA's parent model, was to achieve a sufficiently high speed of operation that the extensive (often trial and error) sequence of runs so frequently necessary in research and analysis would be economically practical. Adaptation of existing models (e.g., LCOM, SAMSOM) was rejected because modifications would have been extensive and costs prohibitive for problems of the size that were contemplated. The AURA program is written in the widely available FORTRAN language and achieves a substantially higher speed by virtue of more efficient processing, and by taking advantage of the dramatic increase in the size of the core storage of modern computers. In its current formulation, AURA makes no intermediate use of auxiliary high-speed storage units (e.g., disks, tapes) except for storing the initial conditions for multiple trials.[1]

In AURA, specified numbers of "pacing items" (known as "vehicles") of various types can be assigned to each unit. The vehicles of a given type at any unit, e.g. battalion, may be supported by a common pool of resources (personnel and equipment) or, as an alternative, the vehicles may be organized into two or three sub-groups (e.g., companies) each supported by its own set of resources. The vehicles, which may be

[1] At present, the TSAR and AURA source codes are virtually the same, although certain program dimensions are different and all output formats in AURA use terminology more familiar to Army personnel.

grouped into platoons or smaller elements, are sent on missions in response to a set of user-supplied mission demands, differentiated by unit, vehicle type, and mission priority; if a unit is not specified, the mission demands are allocated to the unit best able to generate the necessary missions. Platoons may be scheduled or initiated on demand, using vehicles that have been placed in a ready status.

Vehicles may be lost on a combat mission, or, when a vehicle returns it may be damaged, still have munitions, and have several unscheduled maintenance task requirements, including a need to be towed before repairs can be made. These unscheduled maintenance tasks are normally accomplished at the vehicle's operating unit, but a vehicle may be moved to a rear location (FAST) for certain maintenance tasks. When vehicles are lost, a replacement may be ordered from CONUS, or, if vehicles are set aside in the theater as fillers, these are used to provide rapid replacements for lost vehicles and, if specified, for vehicles moved to the rear for maintenance. When filler vehicles replace losses, a replacement for the filler force is ordered from CONUS, if such resources are available.

When an operating unit's support area has been damaged by enemy attack, vehicles scheduled to return there may be diverted to other units, preferably to one that normally operates the same type of vehicle. If unit mission-generation capabilities are assessed daily (an option), the unit best able to support the vehicle is selected. As long as a support area remains closed, that unit's mission demands are allocated to adjacent functioning units with the appropriate type of vehicle either in proportion to the vehicles available or, if unit capabilities are assessed daily, in proportion to the mission-generation capabilities of the units. When a support, e.g., task force trains, area has been reconstituted, that unit's vehicles receive maintenance and supplies at that location on completion of their next mission, if unit mission-generation capabilities are not assessed or, if they are assessed, when the reconstituted support area can support a given level of activity.

The next assignment for each vehicle is selected tentatively when it returns; selection takes into account the known demand on that unit for missions and the projected capability of the vehicles at that unit

to meet those demands. The selection also takes into account which of that vehicle's unscheduled maintenance tasks need to be accomplished for the different missions, and when that particular vehicle could probably be readied for the different missions. All tasks that are not essential for the tentative mission assignment may be deferred and the available resources concentrated on required tasks. If vehicles are eventually found to be unneeded for the mission for which they were readied, they are reassigned and reconfigured for a more appropriate mission.

On-vehicle maintenance tasks may require a number of maintenance personnel, specialized support equipment, and spare parts; each task is either a single set of such requirements--i.e., a simple task--or a network of tasks, each with its own demand for personnel, equipment, and parts. When resources are limited, those vehicles most likely to be readied first (given sufficient resources) may be given priority. The basic input data that govern the probabilities for unscheduled maintenance tasks (other than battle damage repairs) may be used directly for the simulation or varied statistically to reflect unexpected differences between planned levels and "actual" wartime experience. Futhermore, these task probabilities--i.e., the breakrates--may either have a fixed rate or be varied daily by shop and vehicle type as a function of achieved commitment rate, or other user-specified adjustment.

If a required part is not available, (1) the broken one that is removed may be repaired at the unit, (2) the appropriate part may be cannibalized from another vehicle, (3) a part may be obtained by (another unit, a FAST, or DISCOM) resupply from a specified subset of nearby units, or (4) the part may be ordered from a central source within the theater. When a part can not be repaired at the unit (i.e., is NRTS) it may be sent to a neighboring unit or to another unit in the support area designated to perform direct or general support maintenance--i.e., a FAST, DISCOM, or COSCOM. When parts can not be repaired within the theater, a replacement may be requested from a depot in CONUS, when specified by the user. Parts may either be a simple part that with some probability can be repaired at some maintenance echelon, or an LRU that has a defective SRU. For simple parts, either one specific procedure is required for repair, or one procedure is selected

at random from two or more repair procedures. For LRUs the resource requirements to diagnose and replace the faulty SRU are specified separately for each SRU. Faulty SRUs, withdrawn from an LRU, may be repaired within the organization or sent to another unit for repair (NRTS).

The various types of support equipment used in on-vehicle and off-vehicle jobs, munitions assembly and loading tasks, and by combat engineers, are themselves subject to malfunction and repair. As with spare parts, equipment repair may follow a specific procedure or may be accomplished by one of several procedures selected at random. The special complexities of full and partial-mission-capability of Automatic Test Equipment (ATE) used to repair LRUs and SRUs for complex vehicles may also be simulated.

Each maintenance task, parts repair job, and support equipment repair job is accomplished by the maintenance personnel and support equipment associated with a particular work center or "shop." The user may group the resources and tasks into up to 25 different shops exclusive of those associated with the scheduled premission maintenance tasks. Since each shop may be assigned several different types of maintenance personnel and support equipment, those engaged in on-vehicle and off-vehicle tasks may be the same or different depending on how the user wishes to define the unit's maintenance policies.

The user is given substantial flexibility in defining the rules by which vehicle maintenance tasks are processed. He may permit the activities of certain groups of shops to proceed simultaneously or may require that the activities of several such groups of shops proceed in a specified order. He also may control these prescriptions for simultaneous and sequential operations separately for each vehicle type at each unit. Furthermore, for those groups of shops that are permitted to proceed simultaneously, certain exceptions may be specified in the form of lists of activities that are incompatible with each particular task. These features permit alternative maintenance operating doctrines to be simulated and to be examined for their influence on mission-generation capabilities. Work speed-up and other procedures to shorten on-vehicle, premission, and off-vehicle activities also may be specified.

Scheduled premission tasks are also associated with the shop structure. These tasks involve vehicle refueling and the loading of both basic munitions and mission-dependent munitions. The likelihood that the basic munitions and the mission-dependent munitions are retained from the previous mission can be specified independently for the two classes of munitions. After mission assignment, vehicle configuration is checked and, if necessary, the vehicle is reconfigured; this may consist of one or two separate tasks, each of which may require accessories, personnel, and support equipment. The loading of the mission-dependent munitions also may involve one or two separate tasks, each with its distinct requirements.

When munitions assembly tasks are simulated, munitions demands are projected periodically to define which types of munitions need to be assembled. Such jobs may require both personnel and support equipment, much like other tasks that are simulated in AURA. Initial stocks of munitions, as well as munitions shipments, are distinquished as to whether the munitions are assembled or not.

Several features permit the user to simulate various "work-around" procedures that can alleviate resource constraints. One such feature permits the user to specify alternative resource requirements for any unscheduled on-vehicle task, parts repair job, support equipment repair job, munitions assembly task, or combat engineering job; one might, for example, specify that a three-man crew could do a normal four-man job in 50 percent more time. Similarly, when accessories or munitions shortages do not permit the normal, or preferred, munitions to be loaded for a mission, several alternative loadings may be specified. A third work-around feature permits the user to designate that certain types of personnel have been cross-trained and may replace or assist certain other specialists. This personnel substitutability feature is operative only for specified units and on specified on-vehicle tasks, munitions assembly tasks, and combat engineering jobs.

The effects of damage due to attacks may be simulated. The user specifies the time and location of the attacks, the repair requirements for the roads and work areas, and the percentage damage suffered by the various resources on the basis of other calculations. (A customized

modification of the AIDA--Airbase Damage Assessment--computer model has been developed[1] that generates and stores unit damage data in the exact format required by AURA.) When vehicles or facilities are destroyed, some portion of the personnel, support equipment, and parts at these locations also may be lost. Vehicles may be sheltered to some degree when sufficient revetments are available, but the vehicle may be considered to be more exposed when certain shop operations are under way at the time of attack; different loss rates are applied in each case. Mission-ready vehicles may be given priority in shelter or revetment assignment, and the damage to these vehicles may be distinguished from that for others. Vehicles in excess of those that may be sheltered sustain a third distinct loss rate. After AURA has decremented the various resources to the extent implied by the damage data, the surviving personnel are reorganized into night and day shifts. After a user-stipulated delay to roughly account for the disruptive effects of the attack, the maintenance personnel resume their activities, unless some specific support equipment like a portable shop van or wrecker is required and has been damaged.

Replacement resources (i.e., vehicles, crewmen, maintenance personnel, parts, munitions, accessories, and building materials) may be ordered from CONUS when losses are sustained. The number of resources that are available for replacing losses may be specified for each type of resource, and the time required to replace the loss may be specified independently for each class of resource.

After an attack, combat engineer personnel, support equipment, and building materials may be allocated according to a priority system to commence the repair or reconstruction of the damaged facilities. Operation of those facilities is resumed when they once again are functional.

In addition to simulating a set of operating units, the user may specify the existence of a corps or theater reserve of filler vehicles, a centralized corps or theater distribution center, or a centralized corps or theater repair facility at which some or all general support

---

[1] D. E. Emerson, *TSARINA: User's Guide to a Computer Model for Damage Assessment of Complex Airbase Targets,* The Rand Corporation, N-1460-AF, August 1980.

(GS) maintenance is conducted. The filler vehicles can, at the user's option, replace losses or replace vehicles that have been withdrawn to a rear unit for maintenance, as well as losses. When additional vehicle resources are specified as available in CONUS, they supplement the filler force. The filler vehicles can be managed according to several alternative replacement policies.

The centralized distribution facility can receive spare parts from CONUS and either retain them until demanded by a unit or transship (some or all) to the unit with the earliest projected requirement. Such a facility can also direct the shipment of parts and other resources from one unit to another. A centralized repair facility, sometimes referred to as a GSRF, is assigned maintenance personnel, support equipment, and spare parts (LRUs and SRUs). Parts are shipped to and from the GSRF from the operating units and are processed in the manner prescribed by the user's choice of which theater management rules are to govern these operations.

The simplest rules for GSRF operation prescribe that faulty parts are repaired in the order in which they arrive, and that they are returned to the sender. The user may also invoke a variety of more complex management algorithms, not only for selecting what to repair and how to dispose of parts when they have been repaired, but for reallocating maintenance personnel, support equipment, and parts among the several operating units. Repair priorities can be based on existing and projected demands and on the relative essentiality of parts for the various missions. Shipment priorities are related to the current and projected demands, retained reparables, and enroute serviceables. When central stocks are insufficient to meet a unit's demand, another unit can be directed to ship the required part, if both the requesting location and the donor location meet certain conditions relative to the importance of the demand and the availability of stock.

Daily estimates can be prepared (an option) of each unit's capabilities for generating different kinds of missions with different types of vehicles. These estimates provide the basis for various vehicle management decisions. One application is in selecting which unit is to be assigned the mission demands for which no unit has been specified. These data can also be used for assignment decisions when

vehicles must be diverted and when vehicles are transferred from unit to unit to cross-level maintenance workloads or consolidate remaining capability.

The theater-wide management of the various resources is supported by a user-specified scheduled transportation system that may be subjected to delays, cancellations and losses. AURA also permits the user to represent a theater-wide reporting system that can be used to provide the central management authority with periodic resource status reports from the several operating units; these reports may be delayed, incomplete, or lost.

When these transportation and communication systems are coupled with the sets of rules for distributing and redistributing resources among the operating units, various concepts of theater resource management may be represented and examined in the context of realistic transportation and communication imperfections. In its current formulation AURA has alternatives for the theater management rules and has been designed to permit additions or modifications to be accommodated readily.

AURA has limitations and omissions that will inconvenience some potential users. The more obvious limitations derive from the manner in which the problem was bounded in designing AURA. Some users will be bothered that AURA treats combat missions simply as delays during which the vehicles are not at an operating unit; others will wish that active bivouac defenses had been included as an integral part of the simulation, rather than being required to consider active defense tradeoffs externally to AURA analysis. Still others will find that these tools would be more useful if the production-oriented, batch processing of spare parts, as they are handled at depots, also were modeled. The ever-increasing importance of treating chemical warfare in military analyses suggests another omission in the current AURA design. Other perhaps less obvious restrictions derive from the absence of time as a variable in TSARINA, and the absence of relative (geographical) locations in AURA.

Each of these design limitations could be a serious obstacle for some potential users, but none of these bounds was accidental or chosen casually. All problems must be bounded, and we believe AURA's

developer's choice of boundaries need not inhibit a wide variety of important and useful analyses. Furthermore, it would be fairly easy--conceptually--to substantially extend or eliminate these boundaries since AURA's existing data structure is sufficiently detailed to be compatible with such additions. Indeed, additions are currently being tested that will permit examination of chemical attacks. But even though such additions are conceptually easy, most would entail difficult design and programming efforts and would further increase AURA's execution time and expand its data collection problems.

The final limitation that should be highlighted is AURA's data input requirements. As one elects to include more and more of the allowed real world considerations, these requirements become substantial. That is not a property of AURA, but of the richness of the user's problem definition; any approach to dealing with his problem at a comparable level of detail would require equivalent information. AURA's main contribution to this dilemma is that it will function comfortably at many levels of detail and the user may quite simply select or reject most of its features and the related data requirements. One important benefit of this flexibility is that analysts can test the potential sensitivity of their results for a particular effect for which the data would be difficult or costly to secure, using invented data that span a reasonable range of uncertainty. If his results are reasonably insensitive to that variation he has a solid argument for neglecting the effect; if they are sensitive he has a compelling argument for mounting the requisite effort to secure the needed data.

II. AURA DOCUMENTATION

This first volume of the User's Manual provides a succinct but complete discussion of the processing logic involved in the twelve major subsets of tasks; eight constitute the simulation proper and the other four deal primarily with housekeeping chores. These twelve are treated in order in Sections IV through XV. Discussion of the input requirements, procedures, and formats are found in Vol. II; these detailed discussions provide the only complete explanations for some of the numerous control options available with AURA and must be considered mandatory reading for any one planning to operate AURA.

The way these materials can be best used undoubtedly will vary widely. If the user's immediate concern is to decide on AURA's adequacy for installation at his organization, his reading should probably begin with R-2769-MRAL. If that decision has been made and the problem is to apply AURA, the user might best begin with a cursory reading of Appendix C of R-2769-MRAL, or this volume, and then turn to the discussion of the input procedures for the sample problem in Section XX of Vol. II. As the user begins to understand how AURA is to be used for his problem and starts to develop the needed input data, he will want to refer to the detailed discussions of the data input procedures in Sections XVIII and XIX of Vol. II. When questions arise as to how AURA will deal with particular aspects of the problem, the user can consult the appropriate section in this volume.

As the user builds his first AURA data base, he is advised to hold down the number of vehicles for his trial problem; that number can easily be increased later. This will minimize the time and trouble to locate, understand, and eliminate the errors that will inevitably creep into a user's first data unit. One to three operating units, with six to eight vehicles per unit, can provide useful and very rapid hands-on experience. And as that first trial problem begins to provide output, the user will want to refer to Section XXI of Vol. II, where the output formats are explained with illustrations from the sample problem.

When all appears to be behaving logically for a simple trial problem, it will be time to explore some of the more complex control variables that the user may elect to apply in his problem; only when those are mastered will it be appropriate to increase the number and types of vehicles.

III. STRUCTURAL OVERVIEW

The complete AURA simulation involves many types of events and many classes of resources as well as a considerable variety of output information. To fully understand the simulation, one must understand what the events are, how decisions are made as to when they begin and when they end, and what resources are required. Of particular importance are the internally generated events that must be defined, initiated, and concluded, and that sometimes must wait or be interrupted. On-vehicle vehicle maintenance tasks, off-vehicle parts repair jobs, support equipment repair jobs, munitions assembly jobs, and combat engineer reconstruction jobs generate such events.

In broadest terms the AURA simulation can be divided into three phases: the input and initialization phase, the simulation, and the output phase. The MAIN executive routine initiates these computational phases and, assisted by the TRIALS subroutine, controls processing for the specified number of trials as suggested in Fig. 1. Each of the three phases uses various subroutines to carry out the required computations.

Figure 2 indicates the interactions among the subroutines that are used to input data and to initialize the various data arrays according to user-specified instructions. Subroutine INIT zeros out the storage space for the named common statements and then subroutine INPUT enters data that describe the resource requirements for the different types of tasks, mission characteristics of the vehicles, unit resource stock levels, descriptions of the intra-theater transportation, and communication systems, and so on. Subroutine WRAPUP manages a series of computations that generate a variety of derivative data used during the simulation. After subroutine INLIST and INITIZ have listed key input data and initialized the dynamic storage arrays, subroutine TRIALS takes over control until the simulation is completed and the final outputs are prepared and printed. Before transferring control to subroutine MANAGE, which manages the simulation proper, subroutine TRIALS establishes the user-specified initial conditions of outstanding on- and off-vehicle

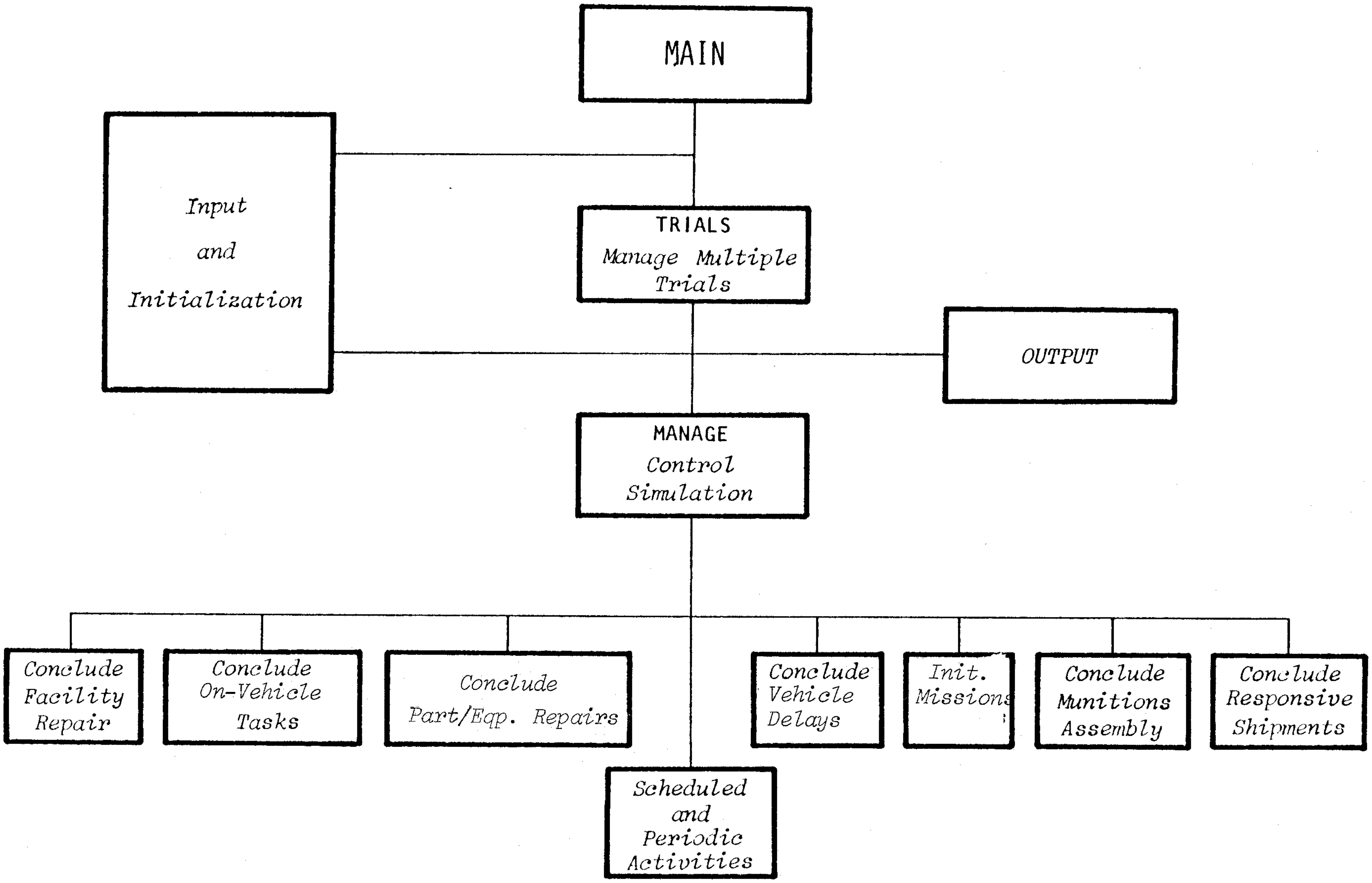


Fig. 1 — Basic structure of the AURA simulation

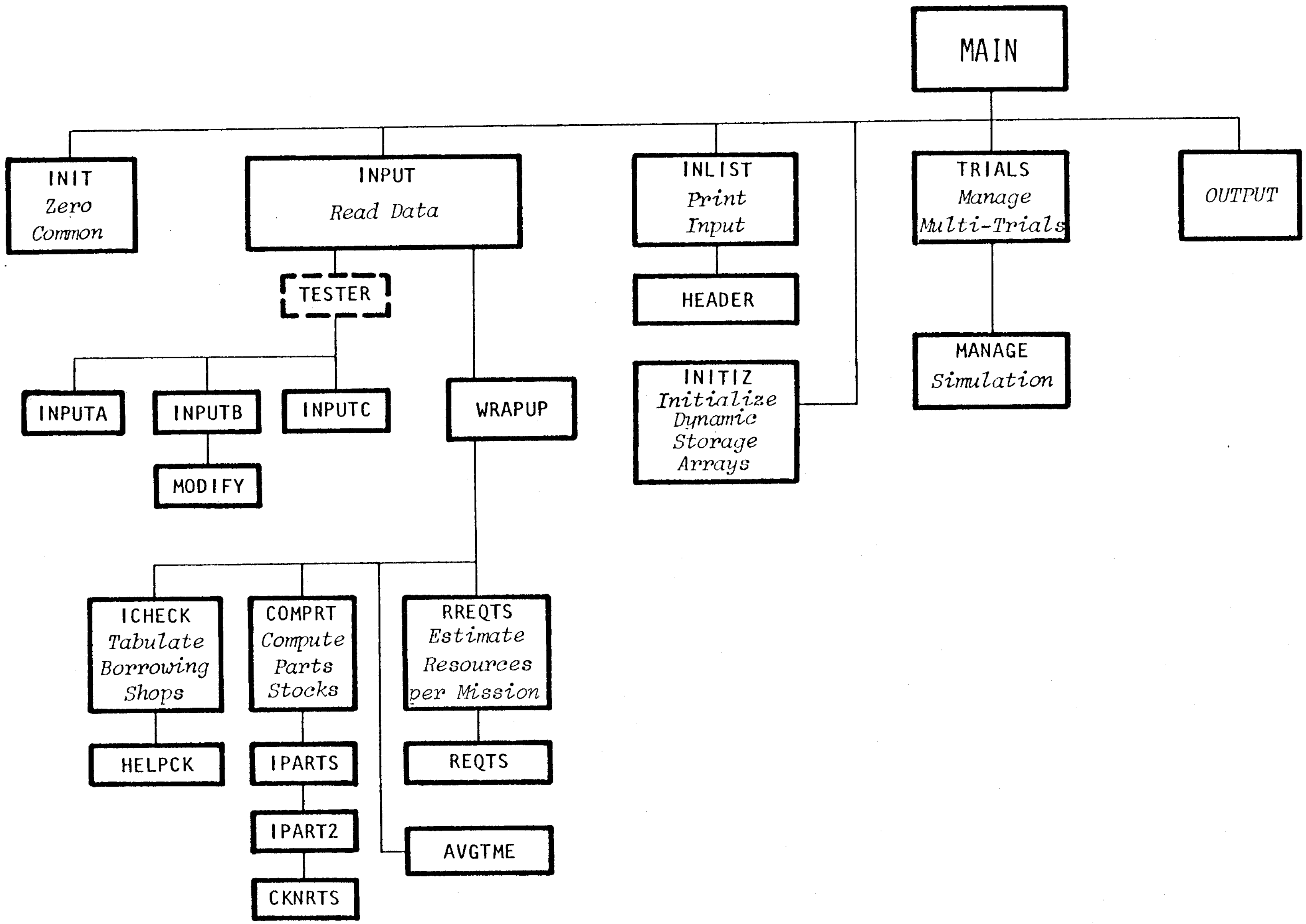


Fig. 2 — AURA input and initialization

work, and reads the first of the mission demand data. Then, as suggested in Fig. 1, control is passed to subroutine MANAGE, which carries out each simulated event in its appropriate time sequence.

Basically, the AURA computer simulation processes one event after another in the order in which the events occur in simulated time, initiating whatever subsequent actions are dictated by the prescribed behavior logic for each type of event. Each of the main tasks indicated in Fig. 1 is performed by a cluster of subroutines supported by a set of storage arrays. Although there is substantial interaction between these tasks and their subroutine clusters, much of the discussion in the following sections will examine one major task at a time, only noting the interactions in passing.

The general organization of these subroutine clusters is indicated in Figs. 3 and 4. As can be seen, each subroutine cluster controls one of the irregularly reoccurring types of operating unit activities; one set is used to initiate vehicles and another is used to process vehicles when they are recovered; others release resources when on-vehicle tasks, parts and support equipment repairs, munitions assembly jobs, and facility repairs are complete. In addition, a variety of scheduled and periodic activities that are necessary during the simulation are processed by the several subordinate subroutine clusters shown in Fig. 5.

To facilitate processing and to avoid the necessity of searching extremely long time-ordered queues, the primary event structure in AURA is divided into the eight different sets of events that have been depicted. Each of these sets is organized so that the next earliest event in each set is always known. Whenever an event is completed, the eight sets are examined in the following order for the next earliest event:

1. Combat engineer reconstruction job completion times
2. On-vehicle maintenance completion times
3. Parts repair and equipment repair completion times

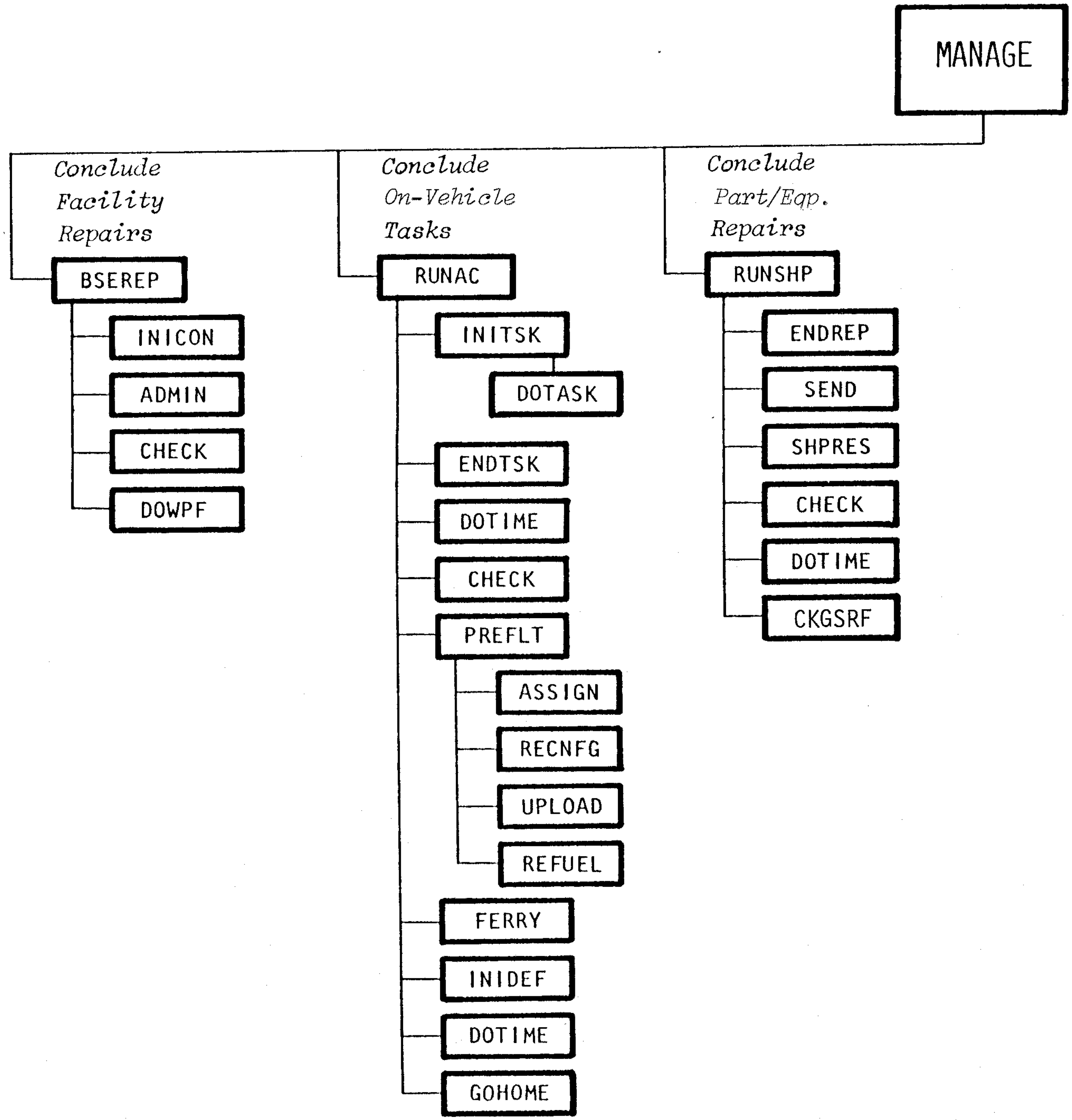


Fig. 3 — Subroutines used to conclude vehicle maintenance, parts and facility repairs

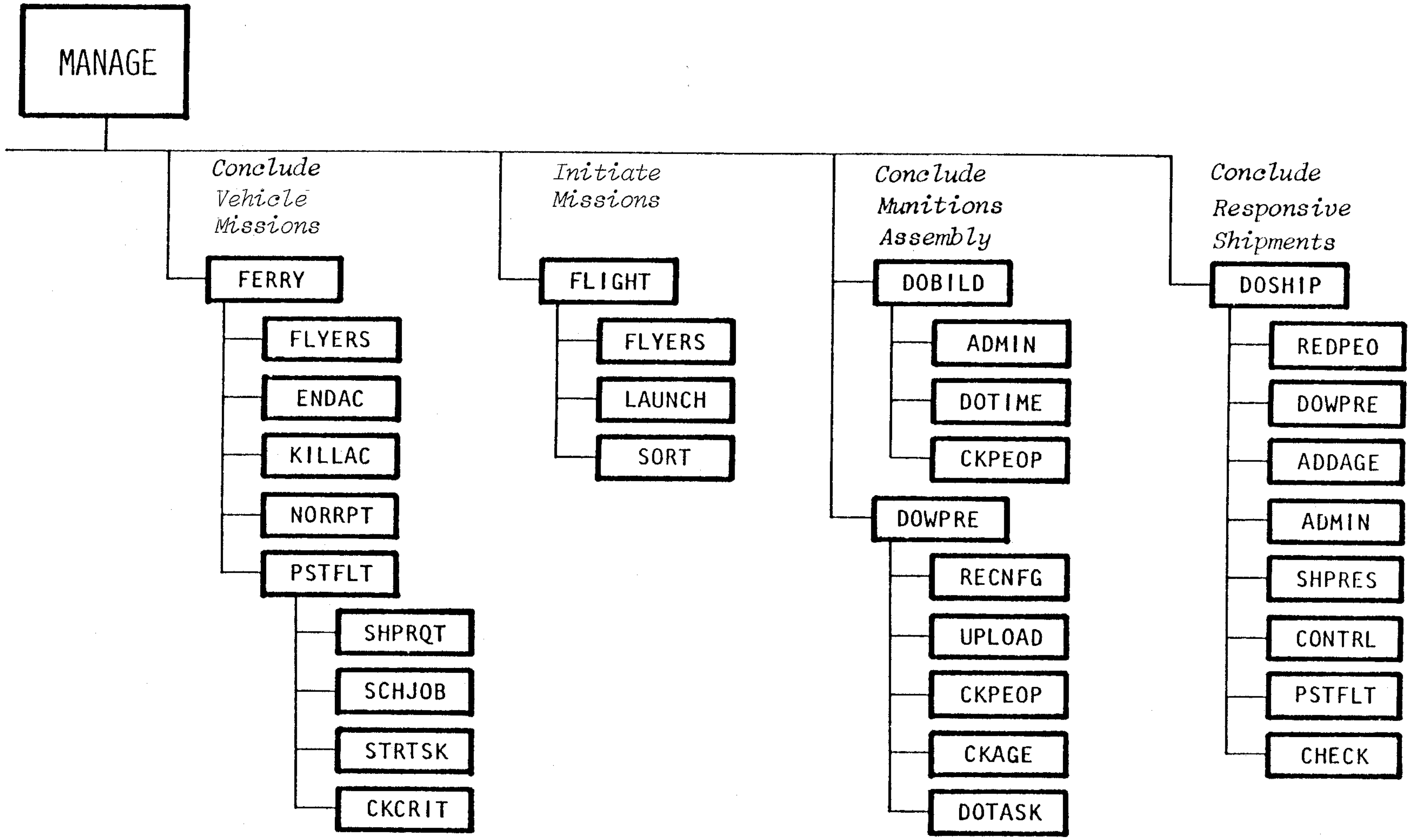


Fig. 4 — Subroutines used to conclude other activities

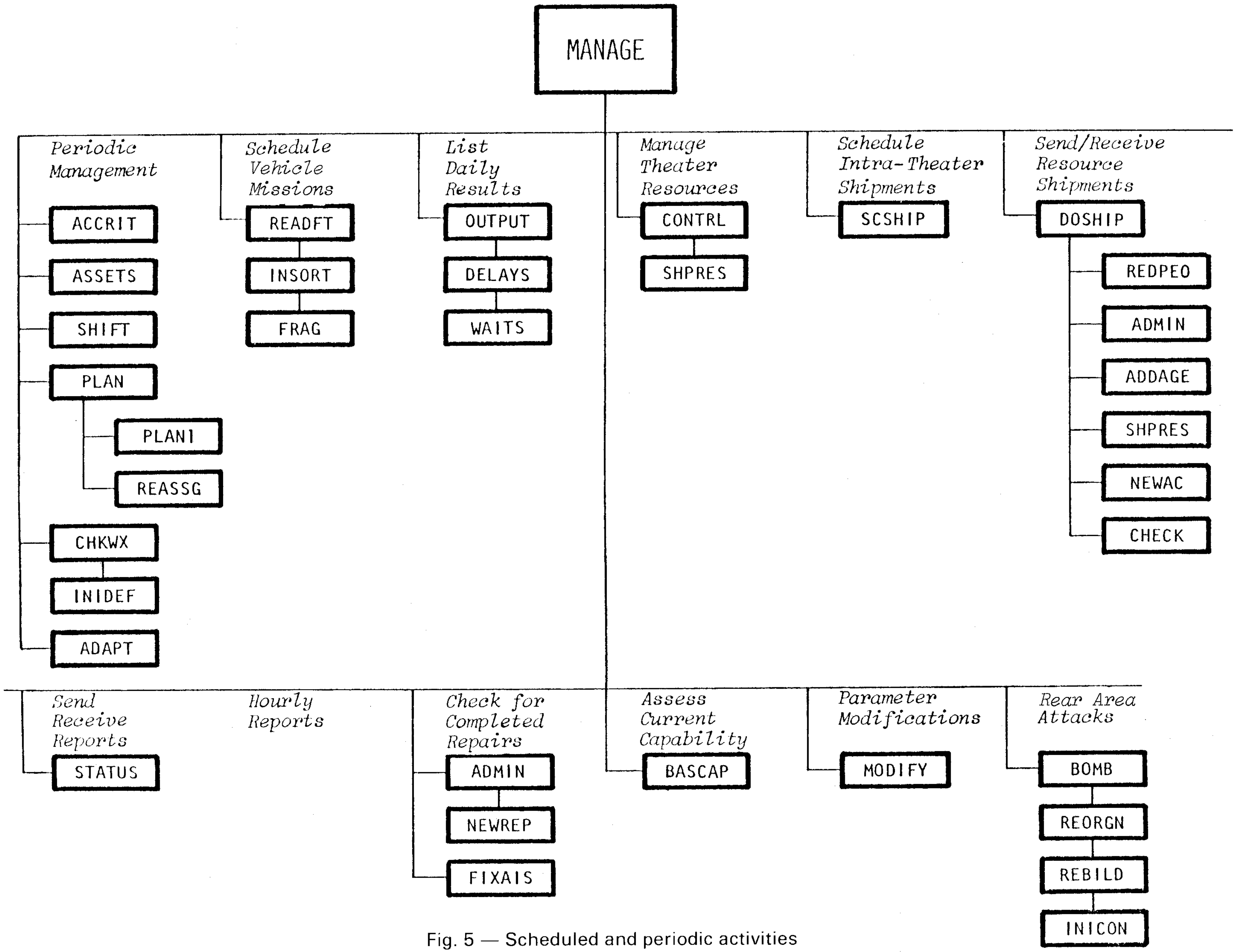


Fig. 5 — Scheduled and periodic activities

4. Periodic and scheduled events
5. Vehicle delay completion times
6. Vehicle mission demands
7. Munitions assembly task completion times
8. Resource resupply arrival times

If two or more of these events occur at the same simulated time, they are processed in the order listed. The order chosen tends to limit the processing requirements.

The nature of these events varies substantially; all except the fourth and sixth sets are groupings of event completion times for similar types of events, whereas the sixth set stores the times at which groups of vehicles (platoons) should be initiated on various missions. The fourth set--the periodic and scheduled events--is a heterogeneous set of events and simulation housekeeping tasks that occur on a scheduled or periodic basis.

Each event set is maintained within a distinct data storage array that also stores other information that must be associated with the event. For seven of the sets, the data are organized into what has been called a "heap," which is a data structure that permits more efficient processing than a time-ordered queue when only the next earliest event must be known. The mission demand data are queued in the order in which the missions will be demanded. In one instance--the periodic and scheduled events--several of the elements in that event set are themselves the earliest elements from subordinate "heaps" and time-ordered queues.

In many instances it is not possible to initiate events as soon as they are defined, or to pursue them without interruption, so it is necessary to store the relevant information until the resources required to pursue the tasks are available. Vehicle maintenance tasks, parts repair jobs, and equipment repairs are stored in special storage arrays if they must wait or when they are interrupted (i.e., the WAITSK and INTTSK arrays); the munitions assembly tasks are stored in the BACKLG array when resources are not available for their initiation and in the INTTSK when they must be interrupted.

The tasks that relate to each vehicle, and to each of the work centers or shops that will perform the work, are tied together with a system of pointers (or storage location addresses). Each vehicle and each shop at each unit maintains pointers to the first and the last of each of these sets of events, and the several events in the storage arrays that are associated with a particular vehicle and with a particular shop are themselves interconnected with a system of pointers. The activities associated with any particular vehicle or shop that are waiting, interrupted, deferred, or in process can be readily located by examining a short trail of these pointers.

The earliest times for each of the periodic and scheduled events are stored as a heap in the array PERIOD, which is maintained in subroutine MANAGE. Some of the events are periodic and some are governed by a user-supplied schedule. At this time there are 15 sets of these events.

Heap Position	Event	Activity
1	Periodic - 2 hours	Change shifts for support personnel Relieve crews Project vehicle supply and demand Reassign vehicle missions Check munitions requirements and initiate assembly Initiate deferred maintenance as permitted List stocks of parts, munitions, and POL (every 6 hours)
2	Scheduled - N days	Read new mission demand data and regenerate the demand queue
3	Scheduled - N days	Regenerate mission demand data queue
4	Periodic - Daily	Print selected daily results
5	Periodic - H hours	Redistribute theater resources
6	Periodic - N days	Regenerate intratheater shipping schedule heap
7	Scheduled (queue)	Receive intertheater shipments
8	Scheduled (heap)	Initiate intratheater shipments

9	Scheduled (heap)	Receive intratheater shipments
10	Scheduled (heap)	Send and receive intratheater resource status reports
11	Periodic - Hourly	List numbers of vehicles waiting by shop, numbers of vehicles with "holes"
12	Periodic - 2 hours	Conclude administrative delays and process faulty parts for repair
13	Periodic - Daily	Estimate mission-generation capabilities for next 24 hours
14	Scheduled (heap)	Modify operating characteristics at a previously specified time
15	Scheduled (heap)	Attacks on support units

Whenever the processing associated with any one of these activities is completed, the next earliest activity rises to the top of the PERIOD heap and is considered in concert with the seven other basic sets of events.

A full understanding of the AURA simulation is provided by a complete description of the steps followed for each of these several sets of events and by specification of the rules, or algorithms, that are used for the decisions regarding the initiation of follow-on actions and the disposition of the resources that are being accounted for within the simulation. These descriptions and specifications are introduced in Sections IV through XI; Sections XII through XV provide an introductory discussion of input-output procedures and other aspects of simulation management.

In the descriptions that follow, all the features and operating modes of AURA are treated. But the reader should be aware that AURA can function usefully in many less complex modes, when that is appropriate. A great many of the features can be dispensed with by simply not entering the pertinent data. At its least complex, AURA would function with one vehicle, one unit, one mission, a mission duration, a turnaround time, and a single periodic mission demand. No resource other than the vehicle needs to be identified.

IV. UNSCHEDULED VEHICLE MAINTENANCE

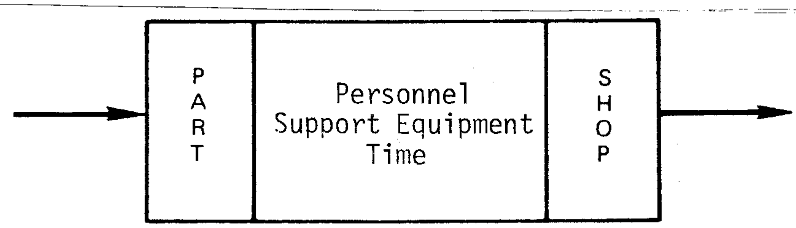
The only constraints on the continuous recycling of vehicles in wartime are the requirements for the availability of crews, munitions, and fuel and the necessary maintenance to permit the vehicle to go on militarily useful missions. Of these constraints, the last is clearly the most complicated since it involves complex interdependencies among a variety of resources. Without maintenance constraints, estimation of a unit's mission potential would be straightforward and would require little or no complex analysis. A basic reason for the level of detail in AURA's formulation was to gain an understanding of the effect of high levels of mission demand and battle damage on the complex processes that are needed to ready vehicles for combat and that depend on several other actions and resources.

Vehicle maintenance can be divided into scheduled and unscheduled tasks. The scheduled requirements include (1) periodic maintenance, performed at specified intervals of time, (2) certain essential support tasks, (3) reloading basic munitions, and (4) premission maintenance tasks (loading mission-dependent munitions and refueling) prior to each mission. As currently designed, AURA does not simulate major periodic maintenance, because it is assumed that such maintenance would be postponed during the critical phases of conflict. The requirements for loading a vehicle's (non-mission-dependent) basic munitions and mission-dependent munitions hinge on the likelihood that the munitions were expended on the previous mission. The other problems that develop and demand attention constitute unscheduled maintenance. Within AURA, unscheduled maintenance tasks develop at random or are generated in battle; the former are categorized as required or deferrable, on a mission-by-mission basis. Deferrable tasks may be completed after some number of missions, before the next day's activity, or they may be deferred indefinitely if mission requirements do not require their completion. For some tasks it may be required that the vehicle be moved to a better-equipped support unit, presumably located further to the rear.

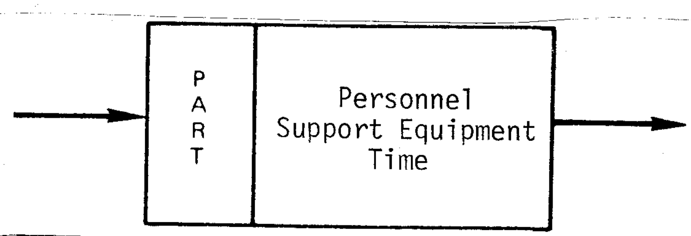
The various specialized personnel, support equipment, parts, and facilities that constitute a unit's maintenance capabilities can be represented in AURA. The maintenance and support equipment may support all the vehicles from a common pool, or they may be organized into subgroups that support subgroups (company teams, batteries, or troops) of vehicles, and a battalion-level organization that supports the several subgroups. User-supplied information describes the various tasks that may be performed on a vehicle (on-vehicle tasks), the maintenance personnel and support equipment required to carry out the tasks, and the work-center (or shop) that is normally responsible for each task. The maintenance personnel, support equipment, and parts that are required for the various tasks also are assigned to a particular shop.

AURA permits the user to define the requirements for each on-vehicle maintenance task as a one-step procedure, a multistep network of subtasks, or a sequence of multistep task networks. The requirements in the one-step procedure--i.e., a simple task--may include a number of each of two types of personnel, one or two pieces of (mobile) support equipment, a part, an undamaged (fixed) shop, and an amount of time (specified by a mean and distribution if desired). More complex tasks that involve differing groups of personnel, equipment, and parts are represented with a task network, or sequence of networks.

To portray these options graphically let us represent a simple task, or root segment, as:

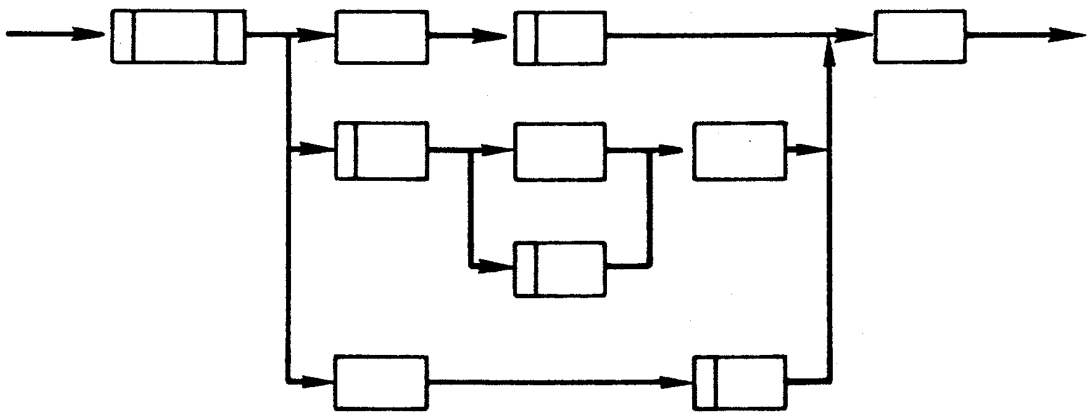


and the other segments of a task network as:



In these terms, on-vehicle maintenance tasks may be represented either as a simple task:

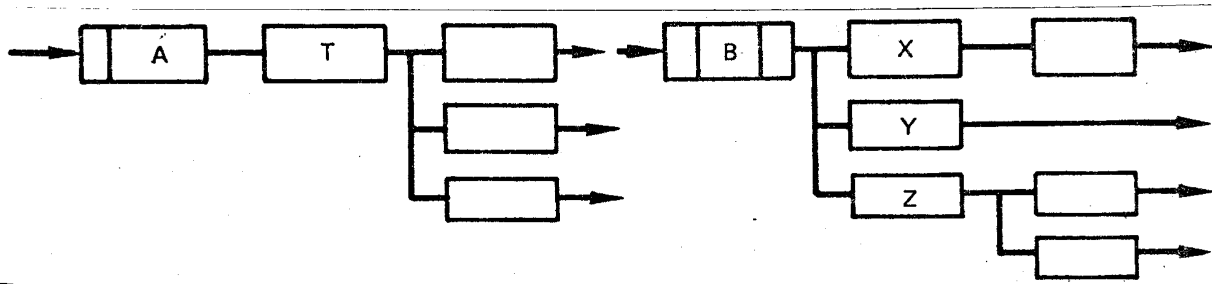
or as a complex network of subtasks:



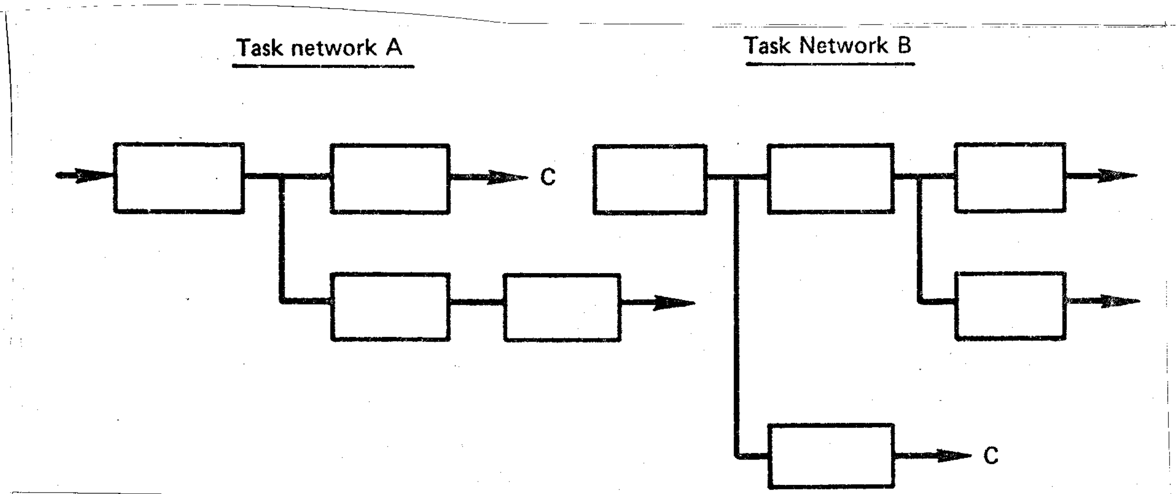
In both cases, if an intact shop facility (fixed) is required to accomplish the overall task, that specification must be included with the first, or root segment. Furthermore, any particular type of part may appear in only one task segment for each type of vehicle. In this illustrative network, when the initial, or root segment, portion of the overall task has been completed, three other subtasks are specified for follow-on activity, followed by yet other activities. The three follow-on activities may all be required, or they may each be required on a probabilistic basis. Furthermore, some of the parallel segments may be defined as being mutually exclusive. If two or more of such parallel paths of activity must be completed before yet another follow-on activity is initiated, this can be represented by those paths rejoining before that subsequent activity. Furthermore it is permissible to represent nested sets of parallel paths that rejoin as illustrated. However, all paths that split and ultimately rejoin must all rejoin at the same place.

The segments of a task network are initiated whenever the resources for the segment are available, without reference to the availability of resources for other segments, unless the "incompatibility" conditions for the segment (see Vol. II, Card Type #19) prohibit task initiation. Although AURA cannot require the time-coincidence of two or more parallel segments that are performed simultaneously in real life, it can be required that all of the segments be complete before any follow-on action is begun, as was illustrated above.

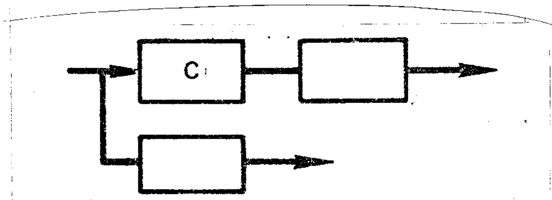
The task segment that immediately follows a segment that includes a part may be made conditional on whether the part was required on that occasion; thus, in the following networks the task T and the mutually exclusive tasks X, Y, and Z may be made conditional on a part having been required in segments A and B.



Task specification and storage may be somewhat simplified when the work procedures associated with several paths have common elements. In schematic terms the two tasks, A and B, can be defined as:



where the C segment



is common to both tasks.

To be able to represent a situation that sometimes occurs in the field, any segment of a network may also specify the root segment of another network as a subsequent task; this simulates the situation where, after work is accomplished on one job, it is discovered that the actual problem is different than initially thought. The only caution to be observed when task networks are "chained" in this manner, is that no two networks may each point to the other, either directly or through intermediate chained networks; otherwise an infinite work loop could be created.

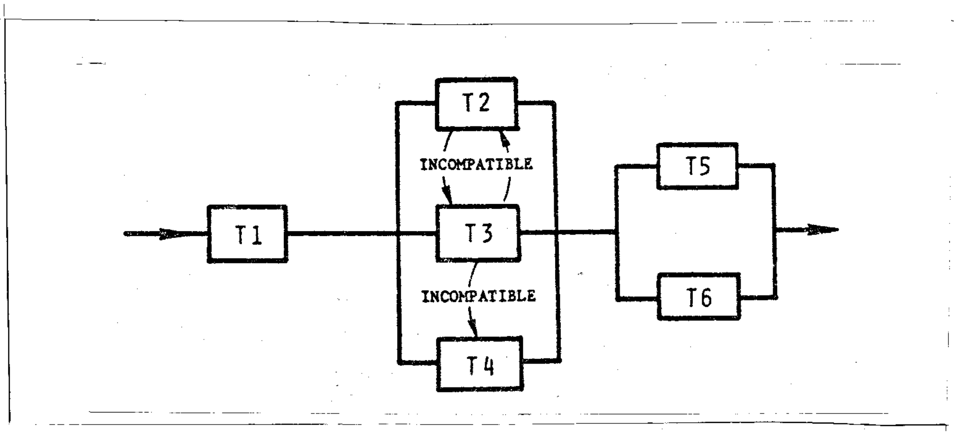
The user may also examine the situation in which certain specialists at specified units have received cross-training so as to be able to assist or replace another specialist on a specified subset of the latter's normal activities. Cross-trained personnel are assumed to perform the tasks for which they are qualified in the same time as the specialists that normally accomplish those tasks.

The user may specify alternative sets of personnel and equipment for any of the segments of a task, and these alternatives will be considered whenever insufficient resources are available to accomplish the task with the normal procedures. If available, the task is done with the alternative resources, without reference to the subsequent availability of the normal resources. There may be as many alternative sets of personnel, equipment, and time specified for each task segment as the user's knowledge and available data permit.

1. TASK ORGANIZATION

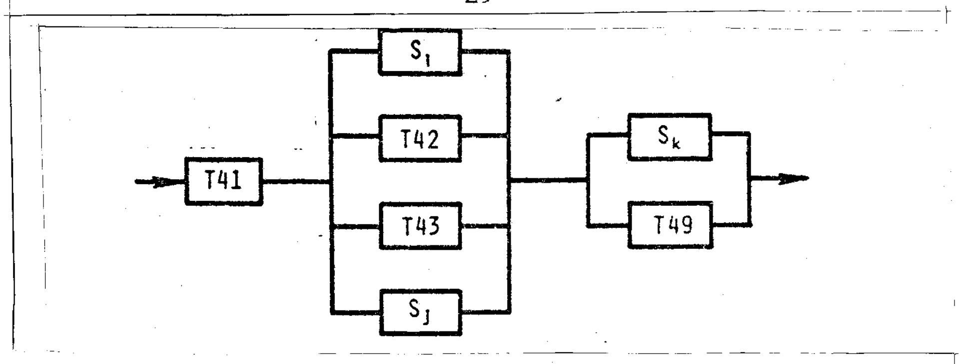
The organization and sequencing of the various tasks that are required on each vehicle are fully controllable[1] by the user for each vehicle type and for each operating unit. Some tasks may be pursued simultaneously, some may have to be done in a specified order, and others may occur in any order, but not at the same time. These options can be illustrated as follows:

[1] Except in the special circumstance in which a vehicle must be moved to a rear unit for some portion of the required maintenance (see subsection 5, Rear Area Maintenance).



In this instance, Task T_1 is accomplished first; Tasks T_2, T_3 and T_4 are done next, as available resources permit, except that if tasks T_2 and T_3 are both required, they may not be done simultaneously, and if tasks T_3 and T_4 are both required, T_4 must be completed before T_3 may commence. Then, when these tasks are all completed, tasks T_5 and T_6 may be commenced; the vehicle may be dispatched when they are all completed. Any or all of these tasks actually may be the root segment of a task-network that must be completed before the task can be considered to be complete. Furthermore, each may occur only with a specified probability. If the vehicle has received battle damage, it can be required that this damage be repaired before task T_1 is initiated, or it may be accomplished at the same time.

The majority of the unscheduled on-vehicle tasks are normally grouped together with the other tasks performed by the same work center or shop for reasons to be described shortly. Reference to these collections of on-vehicle maintenance tasks simplifies the specification of task organization as illustrated in the following sketch.



Here S_i, S_j and S_k are the collections[2] of on-vehicle tasks associated with shops i, j and k. Following a vehicle mission, each collection is checked to see whether any task associated with that shop is required. Since the majority of the unscheduled on-vehicle maintenance tasks are, individually, low probability events, AURA groups together those tasks performed by the same work center or shop and selects at most one following each mission. Processing is speeded up by this simplification (of at most one task per shop per mission) without a serious loss in fidelity, since the joint occurrence of two or more individually low probability events would be quite unlikely.

2. TASK DESCRIPTIONS

The descriptions of on-vehicle maintenance tasks fall into four categories: (1) unscheduled maintenance tasks included in a shop-task-collection, (2) premission tasks, (3) battle damage repair tasks, and (4) other on-vehicle tasks. With the exception of the premission tasks (to be discussed in Section VI), the data defining the personnel, equipment, parts, and time required for each task (and for each segment of task networks), along with the data defining the network structure and parts requirements, are stored in the TSKRQT array. If special damage repair personnel (e.g., contact teams) are to be used for repair of battle damaged vehicles, that requirement can be imposed simply by identifying such personnel as a unique type.

Data defining the likelihood of these tasks are handled differently for each of the four categories. The likelihood that one of the tasks

[2] Up to NOTASK tasks may be grouped together in each of these collections.

in a shop-task-collection is required is stored in the SHPRQT (shop requirement) array (using data input with Card Type #7). If desired, these break-rate data may be varied statistically from the input values for use in the simulation--to represent uncertain wartime break rates--or they may be varied with vehicle activity rate for specified shops and types of vehicles (see Card Type #44).

The vehicle repair requirements imposed by battle damage are handled somewhat differently. Following each mission a random number is compared with the probability that that particular type of vehicle will be damaged on that type of mission (as specified by the user using Card Type #16). For those vehicles that are determined to be damaged, each of the battle damage tasks specified for that vehicle type is checked; the likelihood that each task is required is specified by the task probability in the TSKRQT (task requirements) array. Vehicle repair requirements imposed by damage inflicted during attacks on support areas are handled in a similar fashion, except that the tasks are added to whatever other on-vehicle work is ongoing at the time of the attack.

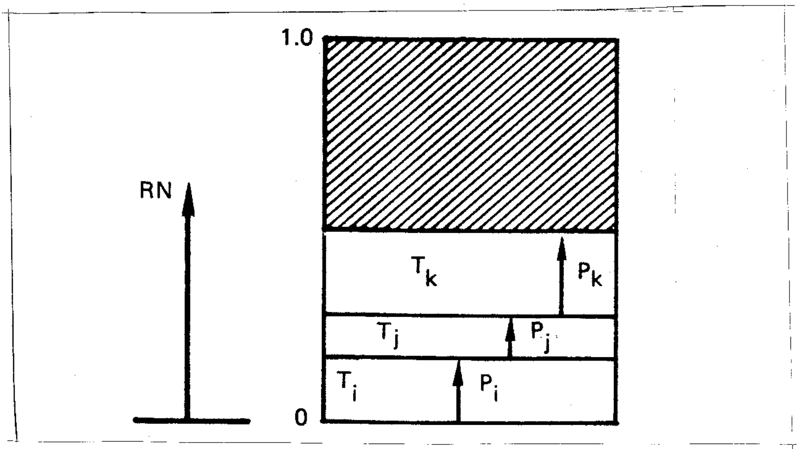
The likelihood that the other tasks are required--i.e., those that are treated individually as tasks #41, #42, and #43 in the last sketch--are specified with the other task specifications in the TSKRQT array, or, for munitions-related tasks, they are controlled by the basic-munitions retention probability entered with Card Type #16. (Such tasks are associated with one of the first 25 shops but do not count toward the limit of NOTASK tasks allowed in a collection of shop tasks.)

With only three exceptions, the requirements for on-vehicle tasks are treated as independent activities. Two of the exceptions concern support equipment, and the other, munitions load crews. For each vehicle type, the user may specify (on Card Type #15) one or two types of support equipment for which multiple demands can be satisfied with a single item. An electric power generator might be treated in this manner. The other exception is used to prevent a single vehicle from being assigned more than one munitions load crew; this feature is invoked when the user specifies the type and number of personnel that make up a load crew on the appropriate Card Type #15.

3. SHOPS

AURA provides for a total of 30 shops or work centers. All vehicle maintenance personnel, equipment, and parts "belong" to one or another of these shops, and lists of the tasks and repairs that are under way, interrupted, waiting, and deferred are maintained separately for each shop. The first 24 shops are used for those collections of unscheduled maintenance tasks performed by specialists associated with each of the vehicle maintenance work centers. If desired, the personnel and equipment of each shop may be assigned to 1, 2, or 3 separate groups (companies) for supporting separate subsets of vehicles. Shops #27, 28, and 29 are used for reconfiguration, munitions loading, and refueling, respectively, as outlined in Section VI. Shop #25, the "ready line" shop, is intended to be associated with those tasks other than the premission tasks that are performed after all or most missions and that may also involve munitions and accessory resources. (Shop #26 is used by the program for storing references to vehicles whose mission assignment and weapons loading has been delayed, and Shop #30 is not associated with vehicle maintenance; it is used in connection with munitions assembly.)

Since, in practice, there is only a limited likelihood that the specialists from any given shop will be required for a task after any particular mission and a much smaller chance that they will be required for two or more distinct tasks, the AURA data structure for the shop-task-collections has been designed such that at most one task from each collection will be selected after a particular mission. With this restriction only a single random number need be drawn and compared with the cumulative sum of the probabilities of the several tasks in each collection. If the number is greater than the sum, no task is required; if it is less, the task to be accomplished is determined by the random number's position within the set of cumulative probabilities. This process may be visualized as follows:



In the situation shown, there are only three possible on-vehicle tasks that the shop may be called upon to perform; T_i, T_j and T_k. The probabilities that the individual tasks are required after any given mission are p_i, p_j and p_k. After each mission a random number, RN, is drawn for each shop. If the value corresponds to the shaded region for a shop, no task is required; if the value is less, the task to be accomplished is the one corresponding to RN. When the user has specified that the nominal task probabilities, or break-rates, for certain shops and vehicle types should be modified in some way (as controlled by Card Type #44 data), the random number is adjusted appropriately before being compared with the shaded region.

These various features for representing the organization and processing of vehicle maintenance tasks will permit the user to rapidly define and test a wide variety of different unit maintenance structures. An example of an actual structure that might be defined is shown in Fig. 6.

Immediately upon return to a support location, the user may specify a finite postmission delay to account for debriefing, etc. This is also the point during the simulation at which AURA determines which tasks are required, what mission the vehicle is to be prepared for, tentatively, and which required tasks may be deferred for the next mission. If the vehicle has suffered battle damage, the repair tasks are scheduled either before any other on-vehicle work or with the first set of on-vehicle work depending upon the value of the control variable CONCUR. In the example, it is specified that Task #45, removing defective ordnance, occurs after 4 percent of the missions. When that is completed, any unscheduled maintenance that is required by Shops #1, 17,

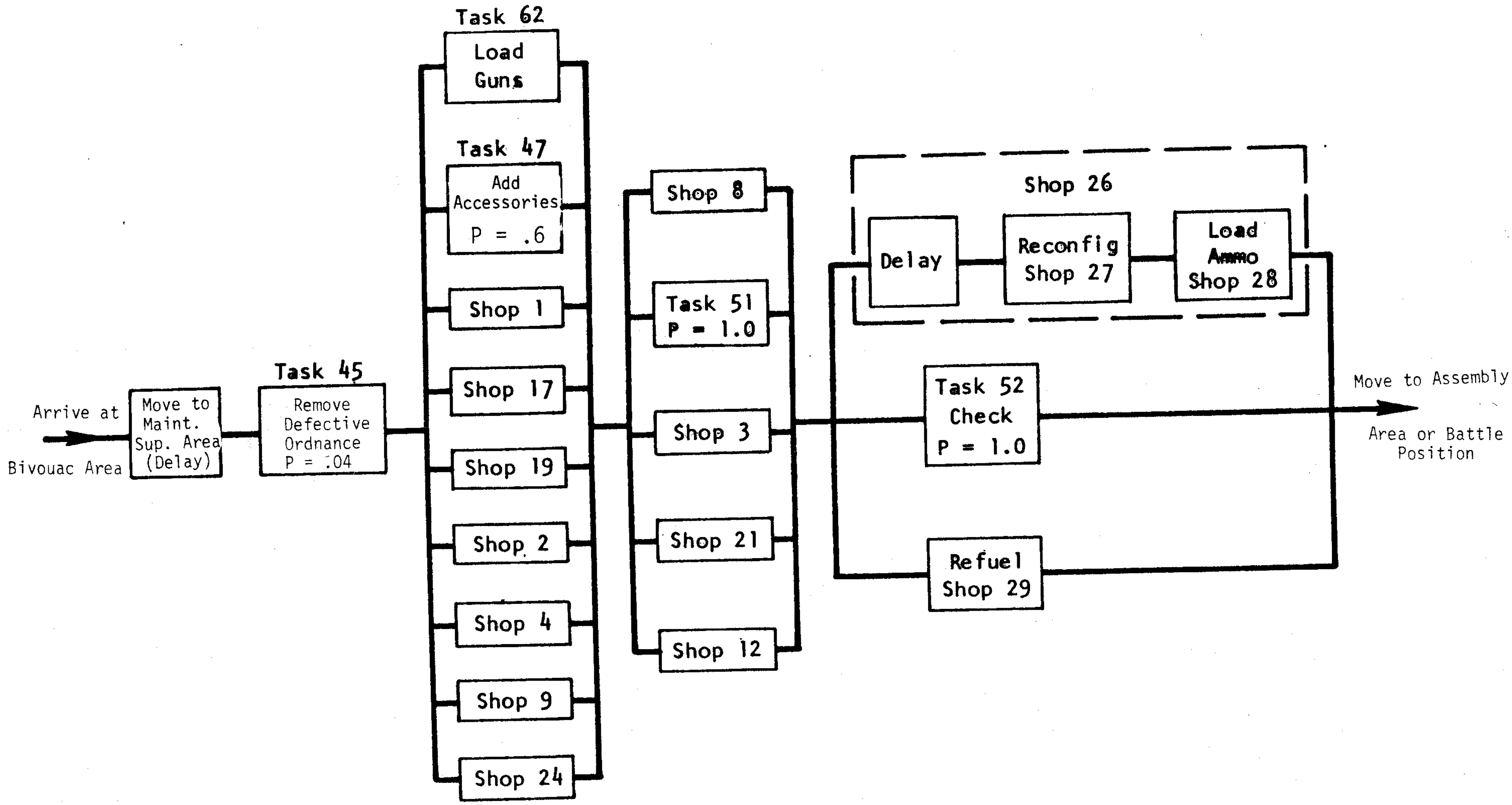


Fig. 6 — Structural representation of on-vehicle maintenance

19, 2, 4, 9 and 24 may be initiated. Two scheduled tasks are also specified: the requirement to reload guns, Task #62, is not mission-dependent and is controlled by the Card Type #16 entry for the basic-munitions retention probability; the task to service rocket launchers, Task #47, must be accomplished after 60 percent of the missions. These tasks are different in character from most other tasks, in that some of the required resources are consumed. Tasks that may consume accessories or munitions *must* be associated with the special "ready line" Shop #25 when they are not part of the mission-dependent munitions activities, or refueling activity in Shops #28 and 29.

When all of the first set of possible tasks has been completed, shop activity by Shops #8, 3, 21 and 12 may be begun, along with Task #51. And when those jobs have been completed the premission preparations may begin. These preparations, discussed at length in Section VI, may involve a possible delay, followed by the final mission determination, vehicle reconfiguration, if required, loading of the mission-dependent munitions, and refueling. As indicated, the delay, reconfiguration, and munitions loading always occur in sequence and are specified by calling Shop #26. Task #52 is also indicated as being accomplished concurrently with the premission preparations. This task as well as the other individual tasks must themselves be associated with some shop (< #25) and use resources assigned to one or another of the shops. The munitions and accessory tasks that must be placed in Shop #25 would use personnel and support equipment from Shops #27 and #28, while the other tasks could call on resources normally associated with any of the first 25 shops.[3]

To specify the task sequence that is to be followed, the user enters a string of numbers using Card Types #29; a different string may be entered for each type of vehicle and for each operating unit. These data are stored in the SHPORD (shop order) array. As the reader may have noticed, individual tasks must always be identified with a number larger than 30, so that they will be distinguished from a shop.

---

[3] Because of the logic used for checking on tasks that are waiting and that may need the resources that are being released from a previous task, the munitions and accessory related jobs--those that use personnel or equipment from Shops #27, #28 or #29--must be associated with Shop #25.

Whenever any of the required maintenance tasks is one of those that must be accomplished at a rear location, or whenever unscheduled vehicle maintenance is estimated to take longer than a user-specified length of time, the user-specified task sequence is replaced by three sequential sets of maintenance tasks. The first set, to be accomplished at the forward operating unit, includes refueling and all tasks that would prevent the vehicle from being moved (i.e., task criticality of 33). The second set includes refueling at the rear location, all tasks that must be accomplished at the rear location, and other of the vehicle's required and deferred tasks as are specified by the variable JOBCON (job control). The third task set, those that are to be accomplished when the vehicle returns to the operating unit, includes all remaining tasks that are required.

4. DETERMINATION OF UNSCHEDULED MAINTENANCE REQUIREMENTS

Whenever a vehicle completes a mission (and has been removed from the delay heap in the ACN array), subroutine MANAGE transfers control to entry point LAND, where checks are made to see whether the vehicle was lost on the mission or has received battle damage. If an operating unit's maintenance personnel have suffered sufficient casualties to be considered ineffective, the vehicles are diverted to another nearby unit for maintenance (see Section XI.1). The crewmen are then released (see Section VIII for a discussion of that process) and, if battle damage is so severe that repair is not practical, the vehicle is written off and the various parts are salvaged to the extent specified (see Card Type #15/2). Otherwise subroutine PSTFLT is called. (For vehicles that have not survived or are salvaged, the existing records are eliminated with subroutine KILLAC and, if replacements are available or if the user specifies replacements, another vehicle is requested using subroutine ORDER.)

The basic functions performed within subroutine PSTFLT are (1) to initiate any user-defined post-mission delay (to account for debriefing, triage, inspection, etc.), (2) to identify what battle damage repairs are necessary, (3) to determine if any deferred tasks must be done at

this time, (4) to identify newly required unscheduled maintenance requirements, (5) to determine whether any of the required maintenance must be accomplished at a rear location, and if so, to schedule vehicle refueling and transfer, and (6) to establish a tentative mission assignment for the vehicle and to categorize the newly defined tasks as essential or deferrable for that mission.

Subroutine PSTFLT establishes which of the individual tasks and which of the collections of unscheduled tasks are required and what the expected time is for carrying out the essential maintenance. When a vehicle has received battle damage, the required repairs are determined, and then the individual tasks and the shop task collections are checked in the specified order as illustrated in the preceding subsection; for each of the tasks and shops a random number is drawn to determine which if any of the tasks require attention. As the various tasks and shops are checked, the expected time to complete each task that is identified is estimated as the mean time specified for that task, plus approximate time allowances for (1) vehicles that are already waiting at that shop, (2) parts repair, when parts are known to be required and none are available, and (3) repair of the maintenance facility itself, when the task specifications prescribe that the facility is necessary to accomplish the task and it is damaged.

When the times for the required tasks and for premission have been determined, the time at which the vehicle could be combat ready again is established for each possible mission type. If any of the previously deferred tasks are now required or are now essential for any of the missions, the time to accomplish them is included on the assumption that they would be processed simultaneously, before doing the other tasks. These ready-to-go estimates take into account the user's specifications as to which shops may perform on-vehicle tasks simultaneously and which (if any) groups of shops must follow other groups. They also take into account only those tasks that may not be deferred for each particular mission. By making the estimates in this manner, the nominal times at which the vehicle could be readied for the various missions will typically differ for the different missions, and these times will also include at least a rough accounting of the queues, parts shortages, or facility damage that might interfere with the preparations for one mission, but not another.

Unless the vehicle is scheduled to be moved to the rear for maintenance, as discussed below, the next step is to determine the highest priority mission that has insufficient vehicles to meet the known demand, between the times that the vehicle could be ready for missions and the time horizon for planning. If the deficient priority is no higher than that for the vehicle's previous mission and occurs no earlier, the vehicle is committed to the same type of mission that it just completed. Otherwise, the vehicle is tentatively committed to that mission with the earliest, highest deficient priority.

The unscheduled maintenance tasks that are essential for the designated mission are stored in the RQDTSK (required task) array and the others are placed in the DEFTSK (deferred task) array. Final bookkeeping in the PSTFLT subroutine includes updating the vehicle's criticality index (i.e., ACN(-,17)), which maintains a record of which missions may be accomplished despite the maintenance that has been deferred.

5. REAR AREA MAINTENANCE

Vehicles may be sent to a rear-area DS/GS maintenance unit either when specially designated tasks must be accomplished, or when the estimated completion time for the required maintenance exceeds a user specified time (MNTLMT). Both regular unscheduled maintenance tasks and battle-damage tasks may be specially designated as requiring action at a rear maintenance unit; these task designations may apply to all vehicles, or only to vehicles at a forward operating unit. Naturally, the criticality of any such task must be such that the vehicle may be moved. If the user has specified a time limit for maintenance at forward operating units (MNTLMT), and the estimated maintenance time exceeds that value, a check is first made that there is at least as much time for the maintenance in the rear as what must be done at the operating unit. If so, and if the time required for the maintenance that must be done at the operating unit is as great as MNTF percent of MNTLMT, another check is made to see that the time for the maintenance that may be done in the rear is at least equal to MNTR percent of

MNTLMT. When this final check is exercised, it provides the user with a somewhat more realistic control over which vehicles are sent to the rear and which are not.

After a vehicle has returned and its ready-to-go time has been estimated in subroutine PSTFLT, as discussed above, a check is made to see if vehicle maintenance is to be done in the rear. If (1) tasks that must be done in the rear are outstanding, or (2) the ready-to-go time exceeds MNTLMT, the required tasks are regrouped into three sets. The first includes a refueling and all tasks that prohibit the vehicle from being driven to the rear. The second set includes a refueling and all tasks that must be done in the rear, as well as whatever other tasks are to be accomplished, as defined by the value of the control variable JOBCON (see Section XIX); these tasks are scheduled for the rear location. The third set of tasks includes a refueling and all other tasks that are outstanding and the necessary munitions loading tasks; these are scheduled for accomplishment on return to the operating unit. No additional maintenance requirements are assumed to be generated on the return trip to the operating unit. To the extent practical, the ordered structure for maintenance tasks that is prescribed with the Type #29 Cards is maintained within each of the three task sets.

Vehicle spare parts for rear-area DS/GS maintenance units are either individually stocked or, when the automatic parts generation feature is being used, they are acquired by redistributing the spares that are calculated for the operating units. For tasks that must be done in the rear, all parts are placed in the rear. An estimate is also made of the fraction of the other tasks that will be accomplished at the rear unit at the same time that the mandatory work is under way, and a like fraction of all parts is placed in the rear. If vehicles are also sent to the rear whenever the ready-to-go time exceeds MNTLMT, etc., the fraction of the parts placed in the rear can be increased by the user's specification of RPARTS.

6. VEHICLE MAINTENANCE MANAGEMENT

After a vehicle's next mission has been tentatively selected and the various scheduled and unscheduled tasks have been defined in subroutine PSTFLT, subroutine MANAGE transfers control to subroutine RUNAC, which manages the initiation and termination of on-vehicle maintenance tasks until the vehicle has been prepared for operation.

When first called, following the optional postmission delay, subroutine RUNAC is entered at entry point STARTM (start maintenance). AURA immediately attempts to initiate the required work on each of the first set of required tasks stored in the RQDTSK array by calling subroutine INITSK (initiate task) through entry point NEWTSK. When a vehicle has sustained damage in battle, those tasks are scheduled first, unless CONCUR is unity. If the required resources are available to initiate a task, they are withdrawn from stock, the task completion time is determined (using TTIME), and the activity is placed in the TASKQ heap; if resources are not available the task is placed in the WAITSK (waiting task) queue of the appropriate shop. (The operation of subroutine INITSK will be discussed more fully in the next subsection). When all of the tasks that may be performed simultaneously have been processed, control is returned[4] to MANAGE for other operations.

Subroutine RUNAC is also called whenever an on-vehicle task has been completed. The first step is to call subroutine ENDTSK to release the resources[5] and to assign them to tasks that may have been interrupted or are waiting (by using subroutines CHECK and DOWPRE for unscheduled and premission tasks, respectively). When ENDTSK returns control to RUNAC the next step depends upon the nature of the task. For unscheduled maintenance tasks a check is made to see if the task is an

---

[4] When late departures are permitted, each vehicle is also checked to see whether its estimated ready-to-go time is sufficiently close for it to be considered. If other tasks remain to be checked, but the estimated time is within two hours, the flag--ACN(-,21)--is set so that the vehicle could be considered for a possible late departure. The flag is also set if all tasks have been started and the completion time is within three hours, or when only one task remains, has been initiated, and is expected to be completed within four hours.

[5] Except for specific types of equipment that may need to be retained for use on other ongoing tasks.

element in a task network, and if it is, resources are checked to start any subsequent task or set of parallel tasks. A check is next made for any tasks that may have been forced to await the completion of the just concluded task, because of an incompatibility (as defined with Card Types #19), and any such tasks are initiated, if resources permit.

If at this point on-vehicle tasks are in process, control is returned to MANAGE; if no tasks are in process but tasks are still waiting for the appropriate resources, a new estimate is made of the ready-to-go time before returning control to MANAGE. If there are no tasks in process or waiting, but tasks remain in the RQDTSK array, a new estimate of the ready-to-go time is computed, and the next set of required tasks is checked and initiated as resources permit. If no tasks are in process or waiting, and there are no further tasks required, a check is made to see if conditions permit deferred maintenance to be accomplished at this time; operations in this circumstance are discussed in a subsequent subsection.

When subroutine RUNAC is called at the completion of a premission task, operation is somewhat different than with unscheduled maintenance tasks. After the resources that have been in use are released (and an attempt to reuse them made with subroutine DOWPRE), subroutine RUNAC calls subroutine PREFLT, which manages the unique task structure used with premission tasks. These operations are described in Section X. When control is subsequently returned to subroutine RUNAC, processing continues in much the same manner as for unscheduled maintenance tasks, except for the task network tests.

One other key feature of the management operation performed in subroutine RUNAC permits the premission tasks to be deferred in certain circumstances, so that the final decisions regarding mission assignment and munitions may be delayed until further information has been received regarding mission demand. When these conditions (as discussed in Section X) have been met and the premission delay flag DELYPF has been set to unity, the mission assignment and munitions loading tasks (i.e., Shops #26, 27 and 28) are allowed to wait while the other tasks are processed in accord with the specified shop-task structure (i.e., the shop sequence data from Card Type #29). When all required tasks are complete, deferred tasks will be initiated if it is estimated that they

can be completed before the user-specified last allowable hour for beginning the munitions loading procedures (i.e., LSTTOD). If none can be started, or if all deferred tasks are completed before LOADTM (the earliest hour for commencing to load munitions), a premission delay is computed such that it will just be completed at LOADTM, and the vehicle is placed in the delay heap in the ACN array.

7. ON-VEHICLE TASK INITIATION

On-vehicle maintenance tasks, except for the premission tasks, are initiated with a call to subroutine INITSK (initiate task). This subroutine is initially called from RUNAC; if tasks must wait or are interrupted after they are initiated, subroutine CHECK subsequently calls to recheck the availability of the required resources.

When subroutine INITSK is entered to check for tasks that have been waiting or interrupted, a rough check is made of the existing ready-to-go time estimate for the vehicle; if it is outdated, a crude update is calculated. When a part will be required, unit stocks are checked to see if one is available. The task is then checked to see if it must be delayed because other work in process is incompatible. If there is no problem, the program next checks for the availability of any facility that may be required and for the personnel and equipment specified for the task. If it has been specified that only one item of the required type of support equipment is needed for several tasks, and one is already assigned to the vehicle, the additional requirement is ignored; similarly, a task requiring a munitions load crew is delayed when one is already at work on the vehicle. If the vehicle is assigned to its own company, battery, or troop with its own personnel and support equipment, the required resources are drawn from the appropriate group. If a facility is needed and it is unavailable, or if insufficient equipment is available, the shortage is noted and the program transfers to check any alternative procedures that the user may have stipulated for this task.

Subroutine GETPEO is called to check on the availability of the required personnel, and subroutine CKAGE establishes the availability of any equipment that is required. If insufficient personnel are

available, but some in-unit personnel have received cross-training, checks are made to see whether such personnel can be used on this task, and, if so, subroutine CKPEOP is called to see if sufficient cross-trained or task-assist-qualified personnel are available. If there are not, but the required number of specified personnel are involved in parts repair activities, those repairs are interrupted to acquire the personnel needed for the on-vehicle task, when the needed part is in stock. The time remaining to complete the interrupted repair is stored with the other repair data in the INTTSK (interrupted task) array. If the required maintenance specialists cannot be obtained by these procedures, the last option is to stop an ongoing maintenance task on another vehicle. This will be done only if the ongoing task has at least two hours remaining until completion and if the vehicle has a projected ready-to-go time at least four hours later than the vehicle for which the personnel are sought.

If sufficient personnel are available, but a needed part is not, a check is made to see if it may be obtained from another vehicle by cannibalization: the various options that exist for cannibalization will be discussed in a later subsection. If a part is not located, the fact that the vehicle has a "hole" is filed by calling subroutine RPTNOR (report not operationally ready); if the rules prescribed by the user permit (see Section XI), an attempt is made to locate the needed part at another location in the theater and to have it shipped.

If all resources are available the task is initiated with a call to subroutine DOTASK that places the task in the TASKQ heap and also places pointers defining its location in the in-process queues associated with the vehicle and with the shop that is doing the work. The duration of the job is determined on the basis of the mean task time and the distribution specified by the user. The user may also represent various actions to speed up and expedite the various maintenance activities by adjusting the appropriate control variables. (See the discussion of subroutine TTIME in Section XIV.)

The DOTASK subroutine is also used when it is necessary to stop an on-vehicle task. Since on-vehicle tasks receive priority over parts repair tasks, the only times that on-vehicle tasks are interrupted is (1) when a task is stopped for a higher priority on-vehicle task, (2)

when the number of personnel at a shop is reduced because of a shift change, or (3) when the unit is attacked and shop personnel are lost. At those times the subroutine is entered at the entry point STPTSK (stop task), and the needed bookkeeping is done on the pointer systems used with the vehicles, the shops, and the INTTSK and TASKQ arrays. When personnel are reduced because of a shift change, the last task that was initiated is the first to be interrupted.

If a part is available, but some other resource prevents the task from being initiated, any alternative procedure (set of resources) for accomplishing the task that has been supplied by the user is checked to see if those resources are available. If they are, the task is initiated using the alternative procedure; if they are not, the task must wait in the appropriate shop's wait queue. If the task had already been waiting, processing is complete. If it is being checked for the first time, subroutine ACWAIT is called to store the relevant data in the WAITSK array; the resource for which a shortage prevented the primary procedure from being initiated is taken to be the reason for the delay. When the task is placed in the shop's wait queue it is placed last in line if ORDWT=0; if ORDWT=1, subroutine INWAIT is called and the task is placed in the shop's wait queue such that the vehicle with the least time remaining before it had been estimated to be ready to go is placed first.

The last step for a task that is being checked for the first time is to dispose of any part that must be removed from the vehicle. If the part is not reparable in the theater, it is eliminated and another may be requisitioned from CONUS. If it is not reparable at the unit where it was removed, it is shipped to whatever location has been specified for its repair (using the SHIPTO array data input from Card Type #34). It may be shipped directly after removal from the vehicle, or it may first have to be checked in-unit.[6] If the part can be repaired locally, it is sent to the appropriate repair facility.

---

[6] Any part, LRU, or SRU with a NRTS rate of 101 is shipped directly after removal from a vehicle (or from an LRU); if the NRTS rate is from 1 to 100, the part must undergo the administration delay before being checked for NRTS action.

The part is removed from the vehicle when the task is first checked, even though the resources were not available to start the on-vehicle task at that time; it is assumed that the overall resource demands for the task are adequately approximated by the task's resource requirements whether they are used then or later. The repair of the part is delayed by a time equal to the sum of the nomimal task time and the backshop administrative delay time (see Section VII).

If the task started in INITSK is a task that had already been started but had been interrupted, or is a task that had already been checked but had had to wait, the necessary bookkeeping for the pointers is accomplished before control is returned to MANAGE.

Of the many data maintained for each vehicle, two are flags used to rapidly identify each vehicle's current status; the first flag (stored in the 12th position of the vehicle array--i.e., ACN(-,12)--defines the vehicle's location within the overall mission cycle, whereas the second flag (ACN(-,16)) defines the degree to which the vehicle has progressed through the several steps in the premission process. The states corresponding to various values of the first flag are:

ACN(-,12)	Vehicle Status
1	In action
2	Inactive for the postmission delay
3	Undergoing unscheduled maintenance
4	Inactive for the premission delay
5	Undergoing maintenance following the premission delay
6	Ready for combat
7	Undergoing deferred maintenance tasks

The several premission states defined by the second flag are outlined in Section VI.

8. CANNIBALIZATION

When a part must be replaced on a vehicle and a replacement is not immediately available, AURA may be directed to cannibalize the needed part from another vehicle in certain circumstances.[7] The rules governing cannibalization are managed by the user with his setting of the control variables CANMOD (cannibalization mode), MXHOLE, DOCANN, DOWNTM, and CDELAY. The basic user choices are (1) whether a part may be cannibalized when there are reparables on hand, and if so (2) which of the vehicles may be considered. The vehicles that may be considered must be of the same type and must also be undergoing unscheduled maintenance. Four possible categories are defined: vehicles with parts missing, whose criticality for the designated mission would not be affected; all vehicles that have parts missing; vehicles without holes, if the criticality would not affect the designated mission; and all other vehicles. If cannibalization is selectively restricted to vehicles in either of the first two categories, the donor vehicles must have at least as many missing parts (i.e., "holes") as the recipient. No matter which category is chosen, vehicles that already have a part missing are checked before the others are checked. Parts not normally cannibalized can sometimes be cannibalized if sufficient vehicles already need the same part. The user may also prohibit cannibalization of the part from any vehicle that already has had MXHOLE parts removed, or whose estimated ready-to-go time is within DOWNTM hours; for vehicles without holes AURA has a built-in minimum constraint of 90 minutes for this time.

These optional constraints are defined for various values of the control variable CANMOD as follows:

---

[7] Parts cannibalization may be selectively prohibited by entering '-1' for a specific part type in the CANNTM array using Card Type #35. If the value is less than -1, the part may only be cannibalized when at least DOCANN vehicles already need that type of part.

CANNIBALIZATION CONSTRAINTS

CANMOD	Cannibalization Permitted with On-hand Reparables	Eligible Vehicles (None with ready-to-go time less than DOWNTM hours > 90 minutes)
0		None
1	No	Vehicles with parts missing whose criticality for the designated mission would not be affected
2	Yes	Ditto
3	No	Vehicles with other missing parts
4	Yes	Ditto
5	No	Vehicles whose designated mission is not affected by part
6	Yes	Ditto
7	No	Any vehicle
8	Yes	Ditto

NOTE: Parts that can be cannibalized only when the DOCANN constraint is satisfied are distinguished by an entry in the CANNTM array that is less than -1.

Cannibalization is accomplished in subroutine CANNIB. When a vehicle is checked for the needed part, the waiting tasks, required tasks, and deferred tasks are checked first, in that order. If the same task is found to be required on the vehicle but the part is not required (i.e., is not broken), it is assumed that the part can be removed; if the part may be required in a subsequent segment of the task network, it is assumed that the part is not available. If the task is not found in any of those categories, the in-process tasks are checked; if the same task is not being processed, the vehicle is considered suitable for cannibalization.

When a part is removed from a vehicle, data regarding the new hole is stored in the NORQ array using the NORRPT subroutine (see Section XIV) and is recorded with the related task data by modifying the number of the task (see below). And when the part is removed from a vehicle for which the related task was not already outstanding, a notice to replace the part must be added to that vehicle's list of required or deferred tasks.

To keep track of the state of a vehicle's tasks and related parts a special scheme was created for numbering tasks. If the basic task number is TASK, the value stored for the task depends upon the task's status as follows:

Value Stored	*Status*
TASK	No part required
TASK + 5000	Part required, not yet recorded in NORQ array
TASK + 10000	Part required, recorded in NORQ array
TASK + 15000	Part required, part removed, not yet recorded
TASK + 20000	Part required, part removed and recorded
TASK + 25000	Replace part only, ignore network, part removed, recorded
TASK > 30000	Premission task

When a part is cannibalized from one vehicle to permit work to be carried out on another, the time required to get the part and to complete the basic task on the receiving vehicle is the sum of the time normally required for that task, plus either the time for cannibalizing a part of that specific kind (from the CANNTM array), or the default cannibalization time; the latter is equal to one-half the true time selected for the task plus CDELAY minutes, as defined by the user with the control variable CDELAY.

9. ACCOMPLISHING DEFERRED MAINTENANCE

On-vehicle maintenance that has been deferred as nonessential for a vehicle's designated mission may be taken care of in four different ways. The first possibility (mentioned in the first subsection) is that a different mission will be chosen for the vehicle for a subsequent mission and the deferred task will become mission essential and be transferred from the DEFTSK array to the required tasks in the RQDTSK array.

The second possibility is that a deferred task may be deferrable only for some number of missions (LTHDEF missions) or until the end of the nominal combat day. In the first instance the task will be redefined as a required task after the LTHDEF mission, and in the other it will be redefined when subroutine INIDEF is called after the end of the combat day, as discussed next.

All deferred tasks are reviewed each evening after the end of what the user has designated as the "combat day"; i.e., after ENDAY. At those times subroutine RUNAC calls subroutine INIDEF (initiate deferred maintenance) when all other tasks outstanding for the vehicle have been completed, except perhaps for the premission task set that may have been delayed until the early morning hours. Subroutine INIDEF is also called at 2200 and 2400 during the night by subroutine PLAN to check for needed resources that may have been released by other tasks.

At these times, subroutine INIDEF redefines as required the deferred tasks that must be accomplished at night, and also attempts to initiate each of the vehicle's deferred tasks, if the nominal time for

the task will permit it to be completed no later than the hour specified by the user as the last time at which the rearming process must commence (i.e., LSTTOD--last time of day). After checking that tasks are not already waiting at the task's designated shop, the INITSK subroutine is called to check on the availability of necessary resources. If available, the task is begun; if not, it is left as a deferred task rather than being redefined as a waiting task, since that status would prevent further actions with that vehicle. When the task data have been filed in the TASKQ array the mission-capable status of the vehicle is updated.

The fourth and least likely possibility for working off deferred maintenance tasks occurs on those days for which the user has specified that either the weather (for helicopter units) or command policy causes temporary suspension of operations at a particular unit for certain vehicle types. Subroutine MANAGE calls subroutine DODEF (do deferred tasks) periodically and the weather (or command) status is checked for each unit and each vehicle type at four hour intervals starting at 0400, when it is presumed that the day's weather conditions will be known. For all vehicles that are otherwise ready to go, subroutine INIDEF is called and that subroutine checks whether available resources will permit that vehicle's deferred tasks to be started and completed by the LSTTOD on the following day. This processing follows the same rules as were described in the preceding paragraph.

V. VEHICLE STATUS PROJECTION

1. INTRODUCTION

It is important that a simulation of forward unit operations emulate at least in a limited way the scheduling and control activities that are carried out by maintenance management at each operating unit in order to utilize the available resources efficiently. Choices must be made as to the tasks to be performed, repairs to be done, munitions to be assembled, and vehicle missions assigned. In the real world, these choices are made in the context of a much richer body of knowledge regarding assets, capabilities, and requirements than is possible (or at least practical) in a simulation. Furthermore, the procedures used and results obtained in the real situation are, at least in part, dependent upon the skill, knowledge, and experience of the particular job control managers available, and vary therefore from one circumstance to another. All that reasonably can be expected of a simulation are mechanisms to allow the user to define broadly differing policies for managing vehicle maintenance and repair jobs and to achieve a degree of efficiency in the utilization of the available resources.

AURA incorporates a variety of features for these reasons, a key one of which is the periodic development of what might best be called the projection of vehicle supply and demand. These projections provide the data base, or context, within which decisions are made regarding vehicle assignments, unscheduled maintenance, and munitions buildup for the subsequent two-hour period.

As is outlined at greater length in Sections VIII (Vol. I) and XIX (Vol. II), the mission demand data specify the operating unit, the vehicle type, the number of vehicles, the mission priority, the receipt time of the demand, and the desired start time. Provisions are included that permit the user to stipulate that a number of vehicles of a particular type be maintained in an overwatch (ready) status, so that they may be employed whenever they are needed for a specific mission. These data provide the information with which the pattern of mission demands is projected.

Since the current status of each vehicle assigned to a unit is known at any particular time, one may also project when missions of various types might be started. These projections are also made every two hours for each unit, each vehicle type, and each mission for each of the several priority levels. By comparing these two projections, vehicle assignments are made to give priority to the more urgent, higher priority demands.

These projections are accomplished in subroutines PLAN and PLAN1 and the essence of the supply and demand comparison is stored in the SORDEF (mission deficiency) array for use as decisions are required. The time horizon for these projections is controlled internally and may be made a function of the time of day: the time to the time horizon is divided into 16 time blocks.[1] The mission demand times and estimated vehicle ready times are placed into that time block into which they fall.

2. PREPARING THE PROJECTIONS OF SUPPLY AND DEMAND

Subroutine PLAN is called by MANAGE on even-numbered hours, and the first step is to estimate vehicle supply. Each unit and each vehicle is checked and the estimated ready-to-go time determines which time block is credited with an available vehicle. The ready-to-go time is determined either by the value that was estimated when the maintenance requirements were determined in subroutine PSTFLT (and occasionally updated) or, for those vehicles that are currently in action, the ready-to-go time is projected on the assumption that the vehicle will be reassigned to the same mission and will spend a nominal amount of time in unscheduled maintenance (as specified by the user with Card Type #15). These data are collected for each mission and for each vehicle type and stored temporarily in the SUPPLY array; they are then converted to cumulative distributions over time, and subroutine PLAN1 is called to project the demand and derive the needed comparisons. The ACA (vehicle

---

[1] The time horizon is controlled either by the user or by the default conditions; as currently written, the default conditions are a planning horizon of 12 hours from midnight to 0400, 8 hours from 0401 till 1600, 20 hours from 1601 till 2000, and 16 hours from 2001 till 2359.

assignment) array is updated at the same time for the vehicles that are currently at the unit and have already been assigned to specific missions.

Subroutine PLAN1 is called separately for each type of mission, each type of vehicle, and each unit. The demands for each such subset are first collected for the highest priority demands--Priority #1--in array DEMAND and converted to a cumulative record in array SUM. The vehicle supply for that mission and vehicle type (that was stored in the SUPPLY array) is then projected ahead on the assumption that available vehicles will be initiated when required for the first priority missions and will return, and be turned around in the nominal mission cycle time specified by the user with Card Type #15. The projected surplus or deficiency during each time interval for first priority missions is then stored temporarily in a local array. This entire procedure is then repeated for each of the lower priorities, with the continuing assumption that all higher priority missions are also executed and subsequently turned around, when sufficient vehicles are available.

Three data elements are then stored in the SORDEF (mission deficiency) array for each of the 16 time blocks. The total demand at all priority levels during and subsequent to each time interval is stored in the first position; the highest priority level at which a deficiency is projected to exist is stored in the second position; and the number of missions expected to be available at the highest deficient priority level is stored in the third position. These data are used in assigning vehicles during the subsequent two hour period.

When these data have been prepared for all units, vehicle types, and missions, four final actions are carried out in PLAN. The first is to check whether the flag that will delay the premission procedure should be set. If the nominal combat day is complete--i.e., it is ENDAY or later--the DELYPF flag is set to permit the premission process to be delayed and deferred maintenance to be initiated.

The next action in PLAN is to collect the total number of known mission demands for each type of vehicle and mission at all units and to store that information (in ACMDTA(12,-,-)) for use in the GSRF repair algorithms. Then, subroutine REASSG (reassign) is called to check whether more vehicles have been readied for a mission than are needed;

if so, they are reassigned to a mission that is deficient. The last activity, conducted at 2200 and at midnight, is to check that all maintenance that had been deferred until night will receive attention.

VI. PREMISSION TASKS AND MUNITIONS BUILDUP

The premission events dealt with by AURA include a premission delay, final mission assignment, vehicle reconfiguration, loading of mission-dependent munitions, and refueling. The basic munitions that are always carried are normally entered separately as individual tasks, as explained in Section IV. Munitions buildup tasks are also discussed. The procedures and resources associated with these events are sufficiently different that nine special subroutines were developed. When the basic control for vehicle maintenance is passed to subroutine RUNAC by subroutine MANAGE, the management of the premission events is further delegated by subroutine RUNAC, as was mentioned in Section V; for the munitions buildup tasks, MANAGE transfers control directly to MUNEED or DOBILD.

Before discussing the various rules that govern these events in AURA, some definitions and conventions are outlined. The premission delay was envisioned as a period of dead-time that the user might wish to specify before the munitions-related events and (typically) subsequent to the completion of the unscheduled maintenance tasks. When it is necessary to delay the premission events until after the expected receipt of mission demand information (as discussed in Section V), the length of this delay is modified endogenously. Immediately following this delay, a final determination is made as to the next mission that the vehicle will execute and a tentative assignment is made to a specific mission, overwatch or ready-to-go force, or set of reserve vehicles. These selections are based on the most recent projections of vehicle supply and mission demand (Section V) and may involve a change of mission from that designated tentatively at the time of postmission "inspection." After AURA determines the appropriate vehicle configuration for the most effective munitions that are available for the next mission, the vehicle is reconfigured, as necessary, and the weapons are loaded if they were not retained from the prior mission.

The periodic projections of vehicle supply and mission demand are also used to generate the demands for munitions buildup. The munitions demands imposed by the missions that are expected to be carried out are compared with the available and in-process munitions, and work is initiated to offset any apparent shortfall. The prescribed procedures give priority to the earliest high-priority missions that have been demanded.

Several AURA work centers, or shops, are set aside exclusively for use with the premission events. Shop #26 is associated with the pre-mission delay and assignment, Shop #27 with reconfiguration, Shop #28 with mission-dependent munitions loading, and Shop #29 with refueling. Shop #30 is responsible for all munitions buildup tasks. As discussed in Section V, the "ready-line" shop, Shop #25, also can be used in connection with the basic munitions and certain accessories.

When the premission events, or tasks, are listed in the user-supplied shop sequence data, as described in Section V, Shop #26 and Shop #29 may be listed in any sequence with the individual tasks and other shop numbers. However, the most logical arrangement would be to list Shop #26 (which implies mission assignment, reconfiguration, and mission-dependent munitions loading) as, or with, the last group of shops. Thus, if one had designated only four maintenance shops, and listed the shop sequence as 1, 2, 3, 4, 29, 0, 26, 0, 0, all tasks would be processed as quickly as resources and task incompatibilities permitted, except for the mission-dependent munitions tasks that would be accomplished last.[1] If, however, the sequence were listed as 1, 2, 0, 26, 0, 29, 3, 4, 0, 0, the work required by Shops #1 and 2 would be completed first, the final mission assignment and weapons loading would be done next, and the work by Shops #3 and 4 and the refueling would be done last. In general it would seem advisable to defer final mission assignment and munitions selection as much as practical, in order to permit those decisions to be made with the most current information possible.

---

[1] A zero must be used to separate shop activities that can be done in parallel from those that are to be accomplished sequentially.

A special control variable is provided to facilitate a separation between fueling and the rearming operations. If NOFUEL is initialized as unity, these operations will not be accomplished at the same time; this constraint overrides any contradictory rule implied by the shop sequence listing.

Management of the premission maintenance tasks is facilitated by a flag that is maintained for each vehicle in the 16th position of vehicle array, i.e., in ACN(-,16). The flag may be set to any of 13 different positions, defined as follows:

	Value When Refueling is:	
Premission Flag ACN(-,16)	*Not in Process*	*In Process*
Premission tasks (Shop #26) have not been initiated	1	8
Premission delay is in process	2	Not permitted
Delay (Shop #26) complete; awaiting assignment, or assigned but awaiting 2nd of Shop #27 subtasks	3	10
Reconfiguration (Shop #27) is in process (one or two subtasks)	4	11
Reconfiguration (Shop #27) complete; one subtask of Shop #28 may be complete	5	12
Munitions loading (Shop #28) is in process (one or two subtasks)	6	13
Premission tasks complete	7	14

As can be noted, refueling (or any other task) may not be carried out during the premission delay.

1. MANAGEMENT OF PREMISSION TASKS

Premission tasks are managed by subroutine PREFLT in much the same manner as subroutine RUNAC manages unscheduled maintenance tasks. When tasks for Shop #26 or Shop #29 are first identified in RUNAC, control is immediately transferred to the entry point PRFLT at the approximate mid-point of subroutine PREFLT. Unless the munitions related tasks (Task

#26 et al.) are to be delayed, or another maintenance task is in process, the premission delay is initiated and the premission flag is updated before control is returned to RUNAC. If the delay is not initiated, the task is stored in the wait queue associated with Shop #26.

When the premission delay is concluded, MANAGE transfers control to RUNAC at RUNAC2, and control is immediately passed to the beginning of the PREFLT subroutine, where the termination of premission tasks is managed. The procedure for terminating the other premission tasks is similar, except that RUNAC is called so that the resources may first be released by ENDTSK; that subroutine, in turn, attempts to reassign the resources using subroutine DOWPRE (do waiting premission tasks), which fills much the same function as subroutine CHECK does for unscheduled maintenance. The primary differences between DOWPRE and CHECK are that the former first checks to see that both subtasks of the reconfiguration and loading tasks are complete before reassigning personnel and equipment, and it does not have any equivalent to the parts repair sections of CHECK.

When control is returned to subroutine PREFLT, an attempt is made to initiate the next premission task unless the preceding task has not been fully completed. Four distinct subroutines are used for vehicle assignment (ASSIGN), reconfiguration (RECNFG), munitions loading (UPLOAD), and refueling (REFUEL) tasks, because of the distinctive characteristics associated with each task. These subroutines are called in the appropriate order by subroutine PREFLT and by subroutine DOWPRE when premission tasks have had to wait.

2. MISSION ASSIGNMENT

As soon as the premission delay is completed, the final mission assignment for the vehicle is made using subroutine ASSIGN. The scheduled ready-to-go time is first interpreted in terms of the 16 time blocks into which the periodic estimates of vehicle supply and mission demand are divided. The highest outstanding priority demand for the mission for which the vehicle had been designated at the time of the postmission inspection is then identified. The process is first to identify the vehicle's lowest permissible assignment priority, the

maximum number that are expected to be ready, and the maximum number of vehicles to be assigned at that priority level using data generated by the look-ahead planning process described in Section V. Next, the requirements for overwatch vehicles, and then the requirements for scheduled missions, are each checked from the highest priority level to the lowest permissible level. The vehicle is assigned to the highest priority demand that has not already been filled.

If the vehicle is not assigned by this procedure to the mission for which it was designated, a check is made to see which other missions it could be readied for, taking into account whatever maintenance has been deferred. This procedure is followed for whatever other missions the vehicle is able to accomplish, until the vehicle is assigned. If it still has not been assigned to an overwatch force or a scheduled mission, it is committed to the mission to which it was tentatively assigned during the postmission inspection and is associated with the other reserve vehicles configured for that mission.

In the event the vehicle had returned from its previous mission with its munitions on board, and it is assigned to a different mission, the munitions are returned to stock without any specific delay or requirement for personnel or equipment. Since the new mission will probably require that the vehicle be reconfigured, it is assumed, in effect, that the munitions downloading is a part of the reconfiguration.

3. VEHICLE RECONFIGURATION

After a vehicle has had its next mission assigned, subroutine RECNFG (reconfigure) is called to check whether the various accessories with which the vehicle was equipped for the previous mission are appropriate for the next mission. If not, they must be removed and the vehicle must be reconfigured.

Before explaining those procedures, we will first review how the appropriate weapons load is determined. For each vehicle-mission combination, the user may specify up to five different standard combat loadings (SCLs); these should be ordered with the most desired munitions first. The characteristics of an SCL include a vehicle configuration (a number corresponding to the entries in the CONFIG requirements array), and one or two sets of munitions, each with a specified requirement for

personnel, equipment, and time. Each configuration, in turn, is characterized by one or two sets of accessories, each with its requirements for personnel, equipment, and time. As with such descriptors in the other kinds of tasks, any of these requirements may be satisfied with a null entry; if, for example, the same crew using the same equipment loads two sets of accessories in sequence, the descriptors for the second reconfiguration task could be limited to the accessory, with null entries for personnel, equipment, and time.

In determining whether a reconfiguration is required and what the new configuration should be, subroutine RECNFG checks first on the configuration of the SCL that is preferred for the assigned mission. The status of the munitions shop is checked if that facility has been specified as an essential resource. If that constraint is satisfied the munitions stocks are checked next. Only then is a check made to see whether the specified configuration is the same as or is different from the vehicle's current configuration. If it is different a check is made to see if either of the two sets of accessories is common to the two configurations; if so, it is presumed that they will not need to be changed. When the new accessory requirements are established their in-unit stock levels are checked. If either the munitions or the accessories required for reconfiguration are not available, the next best SCL is checked. If these resources are insufficient for all SCLs, the task must wait. The task must also wait when there are sufficient of these resources for an SCL, but insufficient personnel and support equipment. Cross-trained personnel may be substituted for the normal personnel requirement on those tasks and units that are specified. When all resources are available the appropriate munitions and accessories are withdrawn from stock, and the time for the reconfiguration task is computed on the assumption that it will take the same amount of time to download a set of accessories as is required to load that set, but that the personnel and support equipment associated with the new set of accessories will perform the job.

4. MUNITIONS LOADING

When reconfiguration is complete subroutine UPLOAD is called to initiate the munitions loading tasks. Since the required munitions were set aside when the requirements for reconfiguration were checked, all that needs to be done is to check on the facility itself, when specified, and on the personnel and equipment required for the loading subtasks. If they are available (substitute personnel may be used when specified) a call to ADDTSK places them in the TASKQ. If they are not available, the tasks are placed in the wait queue for the munitions shop; that queue is checked by subroutine DOWPRE whenever resources from that shop become available. If only one of the subtasks can be initiated, the other is placed in the wait queue.

5. REFUELING

Refueling is included among the premission tasks but does not have a rigid relationship to the other premission tasks, as they do with each other. Refueling is accomplished by Shop #29, whose position in the shop sequence list is under the user's control, as discussed earlier. Thus, it may be placed last or first, or grouped with other shops. Furthermore, the refueling task may have its own list of incompatible tasks, as does an unscheduled maintenance task. In addition, the user controls the special variable NOFUEL, which prevents fueling when any of the munitions-related tasks are in process if it is initialized to unity.

Management of these restraints is handled by the PREFLT subroutine and, when necessary, by the DOWPRE subroutine. When conditions permit, subroutine REFUEL is called to process the fueling task. The only feature unique to this task is the requirement for a quantity of POL. The amount of fuel required is taken to be a characteristic of the vehicle type; the other resources required for refueling are stored in the TSKRQT array, along with those for the unscheduled maintenance tasks. When subroutine REFUEL is called, the required POL is withdrawn from stocks and the necessary personnel and equipment are assigned; if the resources are insufficient for the basic refueling procedure, and for any alternative procedures that are listed, the task is placed in

the refueling shop's wait queue. Control is returned to subroutine PREFLT.

6. MUNITIONS BUILDUP

Although munitions buildup is discussed here in connection with the other munitions-related activities, it constitutes a completely distinct set of off-vehicle functions that are managed independently from the vehicle-related tasks in a separate set of subroutines. Resource requirements for the buildup of each type of munition requiring assembly are specified in much the same manner as simple parts repair jobs, but the procedures used to schedule and prioritize these assembly activities are unique to these tasks.

The periodic vehicle supply and mission demand projections provide the basic "operations" data that drive the weapons buildup selection and prioritization logic. Immediately following that projection, subroutine MANAGE transfers control to subroutine MUNEED (munitions needed) to determine munition needs (when the control variable BUILD has been initialized to 1). A tally is first prepared for each unit of the number of munitions assembly tasks that are expected to be completed within the next two hours. Another tally is made of all on-hand munitions that are loaded, assembled, being assembled, or are already waiting to be assembled. Subroutine MUNEED then tabulates the projected missions in terms of start time, priority, mission, and vehicle type, on a unit-by-unit basis. Starting times within the planning time horizon are divided into four time blocks. Demands for overwatch vehicles are presumed to generate equivalent munitions demands in the first and third time blocks.

With these demand data, control is then transferred to CKBILD (check buildup requirements). This subroutine first converts the mission demands into the munitions demanded by the preferred SCL for each particular mission and vehicle type, and then checks whether sufficient munitions are available or committed. The checks are made first for the highest priority missions in the first time period, then for the next priority, etc. Following that, the demands in the second time block are checked, etc. Whenever sufficient munitions are not available or have not been scheduled to be built, a weapons buildup task

is defined--if sufficient unassembled munitions are available--and control is transferred to subroutine DOBILD, where the required personnel and equipment are checked (substitute personnel types may be designated). If tasks cannot be initiated they are placed in the wait queue in the BACKLG array, until the number waiting equals the number of tasks that are expected to be completed before the munitions requirements are checked again. If sufficient unassembled munitions are not available, the adequacy of munitions for the next lower priority SCL (for that particular mission and vehicle type) is then checked. If no munitions can be located, the demand is dropped. This process continues for all priority levels and time blocks, for each unit in turn. Buildup demands generated by missions in the third and fourth time blocks that cannot be initiated are dropped on the premise that they will be reexamined in the next two-hour review, and need not be backlogged at this time.

If the munitions assembly resources are not fully committed to the immediate demands, they may be used to build up a reserve; the choice of the munitions to be assembled is based on the existing supplies and the history of the demands for munitions.

When a munitions buildup task has been completed, subroutine MANAGE transfers control to the ENDBLD entry point in subroutine DOBILD, where the task is removed from the BUILDQ heap, the shop pointers are updated, and the personnel and equipment are returned to stock.

When control is returned to MANAGE it is immediately transferred to the DOWBLD (do waiting build-up) entry point in subroutine DOBILD, where a check is made to see if the released resources can be used for another weapons assembly task.

VII. PARTS AND SUPPORT EQUIPMENT REPAIR JOBS

AURA provides the user with features that permit the examination of a wide variety of questions related to parts stockage and parts repair policies. Indeed, a variety of questions concerning autonomous and consolidated parts repair capabilities within the theater were central in shaping AURA's theater characteristics. In its present form, AURA may be used without any consideration of vehicle parts, or with autonomous operating unit parts repair facilities, or with repair in whole or part at several rear-area units (i.e., a FAST), or with a centralized parts repair facility in the theater, or with a combination of the last three modes. The constraints imposed by faulty support equipment may also be reflected.

A specialized set of subroutines handles the several elements of the parts and equipment repair procedures. The first three of these subroutines can be used to initialize the parts stockage data and the spare-parts pipelines from CONUS to the theater, and, when there is a GSRF, between the GSRF and the forward operating units. The first subroutine used for parts and equipment repair determines the appropriate administrative delay to simulate before initiating the repair process. Following that delay, other subroutines check on the availability of resources, store the repair jobs that are initiated, and conclude the repairs; another special subroutine is available to disassemble LRUs to obtain SRUs. When parts repair is done at a GSRF, other subroutines come into play. These procedures will be outlined briefly later in this section and discussed more completely in Sections X and XI.

1. INITIALIZATION OF PARTS INVENTORY AND PIPELINE DATA

Although the user may enter the initial parts inventory and pipeline data for each unit, much as for the other classes of resources, he instead may elect (by initializing OUTFIT) to have those data generated as an integral part of the input and initialization process. When this option is elected (for some or all units), the nominal

quantities of parts that should be procured for a user-specified number of combat days are determined as a function of the expected demand, order and ship times (OST), safety levels, NRTS rates, repair cycle time, and if desired, item cost.[1]

After all data have been entered, subroutines COMPRT (compute parts) and IPARTS (initialize parts) are called by subroutine WRAPUP to carry out these computations if the control variable OUTFIT is not zero. The estimates are made on the basis of (1) the parts-procurement-policy planning factors that the user enters using Card Types #23/70 and #23/72, (2) the expected daily demand rate for each part based on the task and parts-repair probability data entered with Card Types #5, #7, and #8, (3) the NRTS data entered for each part with Card Type #23/20x (and #23/30x), and (4) parts cost data entered with Card Type #23/66. If desired, the user may specify different safety stock factors for LRUs and SRUs, and for those tasks that may be deferred indefinitely and those that may not.

If the user desires to define parts shortfalls over and above those that are in the pipelines, three options are provided. In the first instance the actual number of each type of part that is procured for a unit can be reduced by a fixed percentage that the user specifies with the control variable SHORT. The other features permit the user to simulate shortfalls differentially for the various part types. Either or both types of shortage may be used to simulate the parts environment that the user judges to be most realistic. The actual shortfall for each type of part will be the expected value of the shortage if RANDM is zero, or will be drawn from a Poisson distribution if RANDM is unity. If NEWPRT is initialized, the parts initialization computations, including these considerations of shortages, are redone each trial.

The number of serviceable items at each unit for each part type is set equal to the number procured, minus the nominal number that would be expected to be in the pipeline. In other words, it is assumed that

---

[1] The user may modify these computed stock levels to reflect stock shortages or expected battle damage, etc., by entering the additional stock with the basic Card Type #23. As now structured, 500 part types may be modified in this manner. The NRTS rate specified with these cards will override any value entered using the #23/20x or #23/30x Cards if the control variable CHNRTS is initialized to unity; a null entry on the basic #23 Cards will be interpreted as a zero NRTS rate.

there are no local reparables. The number in the pipeline (i.e., being repaired elsewhere) is the largest whole integer in the value developed in the prior computation, or, if RANDM is unity, a number drawn from a Poisson distribution with a mean equal to that value. If the number estimated for the pipeline is larger than the number that had been procured (taking shortages into account), the pipeline number is either reduced to the number available, or (when ZNORS = 1) the difference is made up by removing the parts from possessed vehicles at zero time (thereby generating NMCS vehicles).

As discussed in Section V, vehicle spare parts for rear maintenance locations are either entered directly (with the basic Type #23 Cards) or, when the automatic parts generation feature is being used, they are provisioned by redistributing the spares that have been calculated for the operating units. For tasks that must be done in the rear, all parts are placed in the rear. An estimate is also made of the fraction of the other tasks that will be accomplished by the rear DS/GS maintenance unit at the same time that the mandatory work is under way, and a like fraction of all parts is placed in the rear. If vehicles are also sent to the rear whenever the ready-to-go time exceeds MNTLMT, etc., the fraction of the parts placed in the rear can be increased by the user's specification of RPARTS.

When the user is examining GSRF operations, other considerations affect the parts initialization process. For the procurement computation the user may (1) neglect the effect of the GSRF on NRTS rates, and (2) ignore any advantages of scale in the SRU computation, or he may take one, or both, into account. These choices are controlled by the value of the control variable OUTFIT. If OUTFIT is unity, the NRTS rates that are used for computing the number of parts to be procured for each unit are those that would apply if there were no GSRF; and the number of SRUs is the sum of those computed for the individual units, even though all the LRUs may be repaired at the GSRF. This mode (OUTFIT = 1) permits the user to stock a set of units at levels identical to those that would be estimated if there were no GSRF. If OUTFIT is set equal to 3 or 4, the procurement computation presumes those NRTS rates that apply with a GSRF (the data entered with Card Type #23/30x); if it is set equal to 2 or 4, the safety factors in the SRU procurement

computations reflect the scale advantages to be expected when the demands for several units are consolidated at a GSRF.

The authorized level of stock computed for each unit assumes that all serviceable LRUs are at the operating units. SRUs, however, are allocated in the same proportions that in-theater work is accomplished on their parent LRU. Thus, without a GSRF, all parts are at the operating units, but when a GSRF is introduced, some of the SRUs will be at the unit and some at the GSRF for LRUs that are partly repaired locally, and partly at the GSRF. When certain vehicle maintenance tasks must be carried out by a DS/GS rear maintenance unit, any parts used with those tasks are emplaced with the rear unit; furthermore, if the user's choice of JOBCON indicates that other tasks are to be accomplished in the rear whenever the vehicle is there, the portion of the parts that are appropriate for the tasks expected to be done in the rear are also retained by the rear unit.

After the nominal parts level and the available number of serviceable parts have been computed and stored for each type of part at each unit, the parts pipelines are initialized. When there is no GSRF, the parts that are in the pipeline are scheduled for delivery within the user-specified order-and-ship time, with the actual day picked at random for each item. When a GSRF is assumed to be present, there will be some items in the unit-GSRF-unit pipelines and others in the GSRF-CONUS-GSRF pipeline. The mean numbers in each pipeline for each type of part are estimated on the basis of user-supplied data regarding the various times and the daily demands generated by the operating units. Items are then positioned in both pipelines for delivery after the simulation is begun.

2. INITIALIZATION OF STOCKS FOR BATTLE DAMAGE REPAIRS

Parts also may be stocked automatically for repairing battle damage sustained in combat operations. The quantities stocked at each unit are based on a specified number of missions for each of a specified number of vehicles, and on the battle damage rate expected, on the average, during the first 30 days of conflict (assuming the various mission types are accomplished equally). The number of vehicles is the initial number at a unit, or, when OUTFIT is not zero, the number of vehicles specified for the spares stockage algorithms. The number of missions is entered with Card Type #15/2.

If the condemnation rate (DISCARD) for parts removed in connection with battle damage repair is unity, the NRTS rate is immaterial. However, if all parts are not condemned, the NRTS rate specified for the same part in connection with normal unscheduled maintenance will be applied. If the same part type does not occur in connection with a regular unscheduled maintenance task, it is then necessary to specify the appropriate NRTS rate with a standard Type #23/20x or #23/30x Card. However, if the quantities are subsequently changed, using a normal Type #23 Card, the NRTS rate entered therewith will prevail for the simulation.

The stocks of these battle-damage spares that are allocated to the various operating units take into account any task specifications that mandate that the task be accomplished at a rear GS/DS maintenance unit. The allocation also takes into account (at least approximately) the likelihood that some tasks normally done at the operating unit will actually be cleaned up when a vehicle is in the rear for mandatory rear-area maintenance.

3. IN-UNIT PARTS REPAIR

Whenever an attempt is made to initiate an on-vehicle task and a faulty part is found (or a faulty SRU is found during the repair of an LRU), parts that are never repaired locally may be declared NRTS immediately; otherwise parts are set aside for a delay time before the actual repair process may be initiated.[2] The delay is determined in subroutine ADMIN and is equal to the sum of the mean time for the on-vehicle task (to simulate the time for removal) and an administrative delay. (The user specifies the mean and distribution for this delay by shop and by unit, using Card Type #47.) When that delay is completed, the NEWREP (new repair) entry point in the INIREP (initiate repair) subroutine is called. (If the variable EXPEDite is initialized, and there are no serviceable parts of the required type, the administrative delay is reduced by 1/EXPED, or to zero if EXPED exceeds 10. This

[2] If the NRTS rate for a part, LRU, or SRU is 101, it is shipped immediately upon removal from the vehicle (or from the LRU); if the rate is from 1 to 100, any decision to ship the unit is made after an administrative delay.

feature permits the user to simulate an organization in which the time required to process a reparable can be expedited when necessary.)

When the entry NEWREP is called, a check is first made to see whether the part will have to be repaired elsewhere (is NRTS), or whether it can be repaired locally; this is done by comparing a random number with the NRTS rate. The resources required for the repair process are determined next. One or more procedures may be specified for each type of part: the first is assumed to apply when it is determined that the part is to be declared NRTS, unless it was NRTS immediately on removal from the vehicle. If the part is to be repaired locally and has two or more possible repair procedures, the identity of the required procedure is determined with a random number using the data provided on the relative likelihood that one or another of the procedures is required. Each parts repair procedure can specify requirements for a number of one type of specialist, one or two types of equipment, and time; if the part is an LRU that may have a defective SRU, each SRU is specified by including it as an additional requirement in a LRU repair procedure.

The next step is to check whether the shop has been closed by attack and, if not, whether the necessary personnel[3] and equipments are available. If they are not, the repair must wait; when the resources are available, parts that are to be declared NRTS are consigned for shipment with a call to subroutine NRTSIT, and the required personnel and equipment are committed for the specified time (the timing error in dispatching the part before the time has expired is neglected for convenience in coding).

If the part is to be repaired locally and an SRU is defective, the faulty SRU is withdrawn and placed in a two hour administrative delay. Then checks are made to see if a serviceable SRU is in stock. If none are available and a vehicle is NMCS for the LRU, the user may specify (by setting CANSRU > 0) that another LRU of the same type may be sought in the wait queue, and disassembled to obtain its serviceable SRUs if it

[3] If the data entries differentiate between organizational and DS (contact teams) repair personnel, and repairs are to be conducted at a rear maintenance unit where the personnel are not organized in that manner, the personnel requirements are interpreted in terms of the equivalent organizational MOSs.

does not require the same SRU--i.e., it may be cross-cannibalized. Subroutine SALVAG searches the wait queue and carries out the cross-cannibalization. If the repair job still cannot be started, because of the shortage of an SRU, it is placed in the wait queue of the appropriate shop. If the user has specified that jobs that must wait are to be prioritized (by initializing ORDWT = 1), the repair job is placed in the wait queue (using subroutine WAIT) according to the value of the variable TIME, where TIME is defined for local repairs of a particular type of part as:

$$\text{Time} = -\,1000 \times A \times I/T \qquad \text{where "holes" do exist}$$

or

$$\text{Time} = 10 \times (1 + S) \times MTBF/I \quad \text{if no vehicles require the part}$$

where S = Number of serviceable parts of this type locally

MTBF = Mean time between failures of this type of part

A = Number of vehicles that lack this part

T = Mean repair time for this part

I = Measure of importance; function of total combat mission types for which part is critical

When resources become available, the part with the numerically lowest value of TIME will receive priority. As an examination will indicate, these relationships place the greatest emphasis on the most needed repairs that can be accomplished most quickly; or, if no parts are in immediate demand, the emphasis is placed on important parts that are most likely to be required next. Other algorithms could easily be inserted at this point in the code, if the user prefers to consider a different set of rules. (The numerical constants in these equations are simply scale factors that help maintain distinctions between the integer values of TIME.)

When the necessary resources are at hand to initiate the job, subroutine DOREP (do repair) is called and the repair job is entered in the time heap associated with the REPQ array. If the part for which the resources have been committed is NRTS, the repair job is flagged by specifying the negative value of the repair procedure. The DOREP

subroutine is also used when it is necessary to interrupt an on-going repair job. When that occurs the job is transferred from the REPQ array to the INTTSK array, the SHOPS array pointers are updated, and the personnel and equipment that had been engaged are released. A special provision is included to terminate a repair for which the part itself was destroyed during an attack on the maintenance support area.

The INIREP subroutine is also used when resources are released and an attempt is made to start parts repair jobs that have been interrupted or are waiting. The resource requirements are checked, and if the job can now be started, the INIREP subroutine updates the various pointer systems related with the INTTSK, WAITSK, and SHOPS arrays.

When the administrative delay for an SRU is completed, entry NEWREP is called and checks are made to see whether it is to be declared NRTS or whether it may be repaired locally, much as for an LRU. Checks are next made to see if the required personnel and equipment are available to start the repair procedure. If they are not, the repair must wait; if they are, the SRU is declared NRTS when appropriate, and the personnel and equipment committed for the specified time, again much as for the LRU.

When a parts repair job has been completed, control is transferred from subroutine MANAGE to subroutine RUNSHP (run shop). The part or rebuilt LRU is put into stock, and subroutine ENDREP is called to release the personnel and equipment and to update the pointer systems used with the REPQ and SHOPS arrays. Unless the special parts disposition logic (see Section XI) is applicable (i.e, unless SHPREP > 1), the repaired part is retained if it was removed locally or returned to the unit where it was removed. When it is retained locally, and when there are vehicles locally that require a part, subroutine CHECK is called when control returns to RUNSHP. If a vehicle is still waiting for the part, the appropriate on-vehicle task is initiated. When the part was not removed locally, or if the special parts disposition logic selects another unit, the part is shipped to the appropriate unit. Similarly, when an SRU repair is completed, resources are sought to repair an LRU requiring that SRU. When control again returns to RUNSHP, subroutine CHECK is recalled with the shop number to be sure that the newly released personnel and equipment are reassigned if they are needed.

4. GENERAL SUPPORT OR DEPOT PARTS REPAIR

When a faulty part is found to be NRTS, a check is made to find where it is to be shipped for repair. Based on the data supplied by the user with Card Types #34, different destinations may be specified for each type of part, subject to the data limitations outlined for that card type in Section XII. If there is a GSRF in the theater, AURA assigns it a unit number MAXB. If a NRTS part is to be sent to a depot outside the theater, the destination should be entered as (MAXB+1)--i.e., one greater than the largest number of units.

For RR items (an item with NRTS = 100) an option has been provided to permit the nominal shipping instructions to be overridden when the number of serviceable LRUs falls below a specified percentage (ADAPTR) of the unit's initial number of LRUs. When this occurs, the list of units specified for lateral resupply is checked to find a location that is able to repair the item (NRTS < 100) and has an undamaged shop. This option can be used, for example, to simulate an adaptive parts repair doctrine that discontinues reparable shipments to the depot and attempts to accomplish the repair in-theater, when parts stocks are low.

A faulty part may also be shipped to another repair location, even though it would not normally be declared NRTS, when the shop in which the repair must be done has been closed by damage from attack. When this occurs the lateral resupply unit list is checked for a unit with the shop open and the NRTS rate for the part in question lower; if a unit is found, the part is shipped to that unit if the two-way shipment time is within one day of the reconstitution time for the damaged shop.

If the part is shipped to another operating unit for repair, the part is treated just like any other job generated at that unit and begins by undergoing an administrative delay. The number of the originating unit is preserved so that the part may be returned when repairs have been completed if the special parts disposition logic is inoperative. Depending upon the NRTS rate for that type of part at the receiving unit, the part could be shipped to yet another unit; if it is repaired at that unit, it will be shipped back directly to the originating unit when repairs are completed, unless the disposition logic is operative and selects a different destination. It is left to

the user to design the Card Type #34 inputs such that a faulty part will not be NRTSed from one unit to another until it arrives back at the originating unit.

If a part is condemned, or is shipped out of the theater, its replacement, when one is specified, is consigned for delivery directly to the unit of origin, even though a GSRF may be operating, unless the control variable CONSIG is initialized to unity. In the latter case, all parts returned from CONUS are consigned to the GSRF for trans-shipment according to the user-specified theater resource management algorithms.

When a part is shipped to a GSRF in the theater, it is subjected to an administrative delay, but is then managed by a different set of rules that govern the priority it receives and its disposition when the repair action is completed. These will be outlined fully in Section XI after the properties of the transportation and information nets used in connection with these operations are explained in Section X. Parts repair times at a GSRF can be modified by the user to account for the different working conditions, using Card Type #48; this modification can be controlled on a shop-by-shop basis.

5. SUPPORT EQUIPMENT REPAIR

Many special kinds of support equipment are needed for the specialized jobs that must be conducted to support the maintenance and supply requirements of combined arms units. Some of these equipments are both complex and expensive; malfunctions are fairly frequent and the maintenance and repair of these equipments constitute an essential set of activities. Such malfunctions and the repair of faulty support equipment may also be simulated in AURA.

Support equipment repairs are handled in much the same way as spare part repairs, and with many of the same subroutines and procedures. However, two quite different representations of support equipment failure and repair are provided by AURA. The simpler representation is used for all equipments other than the ATE (Automatic Test Equipment)--those complex equipments that are used to test and repair electronics and some electromechanical equipment. The basic distinction is that in the simpler representation, support equipments are either up

or down; ATE equipment, however, may be partially mission capable. Both representations are described below.

Supply Equipment Repairs Other than ATE Sets

Whenever a task that has used support equipment (other than an ATE set) has been completed, each item of equipment is checked to see if it needs maintenance by comparing a random number with the probability that that type of equipment will require maintenance following a job. If maintenance is required, the equipment first undergoes an administrative delay, much as for spare parts, although the length of such delays is different than for parts. When that administrative delay is completed, the attempt to initiate the repair is processed in the same subroutines as a faulty vehicle part. As with parts, each type of equipment is associated with a particular shop, and the repair procedure may either be specific, or be chosen at random from among a set of alternative procedures. Equipment repair procedures specify a type and number of personnel, one or two pieces of repair equipment, and a duration; and, as with other kinds of simulated tasks, alternative procedures may be specified for consideration when the normal resources are not available. But these specifications do not include the spare parts that might be needed to repair the equipment; such problems can be approximated, however, by specifying that equipment repairs can be carried out without delay for parts on some occasions, while on other occasions are subjected to a delay equivalent to the order and ship time for spares.

If resources are available when an equipment repair is first attempted, the resources are assigned to the repair, the completion time is established, and the job is placed in the repair queue, REPQ; if resources are not available, the job must wait. Support equipment repairs that must wait are currently treated in a first-in, first-out (FIFO) priority; if support equipment and parts are competing for the same repair personnel and/or equipment, the equipment repairs are given priority over spare parts for which serviceables are available, but must follow the repairs for parts needed for work on combat vehicles. As currently structured, all support equipment repairs are performed in-unit; equipments are not "NRTSed" to other units.

Simulation of ATE Maintenance and Repair

The specialized support equipment used for testing and repairing the ATE also may be simulated in AURA. A full "string" of ATE may have several different complex electronic test equipments, or "stations," with each type of station used for testing several different LRUs. Each station is composed of many hundreds (perhaps thousands) of submodules, and these stations are themselves subject to various malfunctions that can require substantial maintenance. Furthermore, when any of the numerous low-failure-rate (and therefore unstocked) ATE parts fails, it is necessary to order one from another location, and that station will then be able to test only some portion of its normal LRUs. Thus a station will be in one of three states: fully mission capable, partially mission capable, or inoperative. If two or more stations of the same type are available, partial mission capability generally can be minimized by consolidating all missing parts at one station.

The manner in which these characteristics are modeled in AURA is adapted from another project at Rand. Whenever an ATE station is used to repair an LRU or SRU, the nominal part repair time is increased to allow for maintenance of the station itself. Since such maintenance may actually occur either before or after, or even during the repair of the part, it is assumed that the part is not released until the overall job is completed. When that time is over, the LRU is released for use and a check is made to see if any piece part needed for maintenance on the ATE was not in stock. If so, the station's residual capability to repair LRUs is estimated on the basis of statistics that indicate how frequently each particular LRU repair capability is lost, on the average, when an ATE part is back-ordered. To do this we imagine that each station is divided into a number of sections, or "trays," one tray for each type of LRU, and when a part is back-ordered the mission capability of each tray is determined on the basis of the statistical experience.

During the simulation, a check is made following each LRU repair to see whether during maintenance on the ATE station it was found to need a part that is not in stock. If one is needed, but there are two or more stations of that type on the unit, it is assumed that the needed part

will be cannibalized from another station, if necessary, and that all missing parts are consolidated at one of the stations. Thus, when an ATE part fails at any station, checks are made for each LRU tray associated with that type of station and a list is generated of all LRUs that cannot be repaired until the needed part is obtained. A sample is then drawn from the user-specified order-and-ship-time distribution, and the appropriate receipt time is entered in the LIMBO array; not until that time occurs is the capability restored for repairing those LRUs.

As will be noted, there are no specific repair procedures or specific personnel or equipment used to repair ATE. Instead, the repair time of each part that is processed is increased to account for ATE maintenance, and ATE repair capabilities are probabilistically curtailed to simulate a shortage of parts to repair the ATE.

VIII. VEHICLE MISSION DEMAND AND CREW MANAGEMENT

One objective of a combined arms unit is to provide combat forces at the time that they are required, and a unit's capability for meeting that objective depends as a general rule upon the pattern of the demand. In AURA, that demand pattern is controlled by the user's input data and the user is provided sufficient options that most plausible requirements should be readily simulated.

A demand specifies the combination of vehicles to be employed, the mission, the mission's priority, and, normally, the unit; it also specifies the number of vehicles to be sent (and the minimum acceptable number), the expected time operations are to be initiated, the time that the unit receives the FRAG (fragmentary order), and the location of tactical objectives (or attack) or final battle positions (on active defense).[1] If desired, the user may also require that a specific number of vehicles be maintained in reserve at a particular unit for overwatch duty or unexpected operations. In addition, he may define a composite mission, made up of several sets of vehicles, each with a differing configuration, as would be required, for example, for representing coordinated attacks by armored, mechanized infantry, aviation, and artillery units.

Except for composite missions and specified reserve forces, it is not mandatory that the initiating unit be specified. If the control variables "STATE" and "SELECT" are both greater than unity, a daily estimate is made of each unit's mission generation capabilities, and these estimates are used to designate a unit for any mission demands for which a unit has not been specified. However, since AURA does not include geographic concepts, such selections are not constrained by distance-to-objective considerations.

For user convenience the demand data may be stated either on a day to day basis or in terms of demands that recur each day with a stipulated probability (or any combination of these techniques). For

[1] Some of these may be specified in a SOC (Specific Operational Capability).

the recurring demands (the periodic demands) the start time may be entered as a precise time or as a time block; when a time block is specified, the program picks a time at random from within the block. Furthermore, a number of such missions (up to 32) may be specified with the same entry; when this is done, the start time of each mission is selected at random from the time block. With these features a few entries suffice to represent a rich and varied pattern of mission demands.

1. GENERATING MISSION DEMAND DATA

The initial day's mission demands are entered before the simulation begins. They are entered after all other data are input, and after subroutine INLIST has provided whatever listings of input data were requested. Subroutine READFT (read mission data) reads and organizes these data with the assistance of subroutine SORT, which orders the missions by their specified start times and manages the pointer system used with the FLTRQT (mission requirements) array. If the initiating unit has not been specified for any of the missions demanded, subroutine FRAG is called to select the unit best able to fulfill the demand, the one with the lowest current level of demand relative to its estimated mission generation capabilities. When all data have been entered, missions with common characteristics (unit, vehicle type, mission, and priority) are interconnected with the pointer system associated with the PTZ array.

The mission demands for the next day and for subsequent days are also managed in the READFT subroutine. These demands are reexamined each evening at 2000 simulated time, when this subroutine is called by subroutine MANAGE. If the user wishes to specify new missions or to change specifications for reserve forces or periodic missions, these data are read at this time. If there is no new information, the following day's demands are based on the periodic mission demands or other mission data submitted earlier. As before, any mission demands for which a unit has not been specified have a unit chosen with the FRAG subroutine, using updated estimates of all units' mission generation capabilities, which are created daily at 1930 by subroutine BASCAP.

If, when the mission demands are organized for the following day, a unit is out of operation because of attack damage, the demands on the unit may be reassigned. If the damage is projected to prevent operation for any part of the following day, and other units have vehicles of the type specified, those demands that are required to be met before operations can be resumed are reassigned either by subroutine FRAG, just as though the initiating unit had not been specified, or, if SELECT is zero, in proportion to the numbers of vehicles at each unit. Demands to be met after the resumption of combat operations are not reassigned. The user must exercise care because, as mentioned earlier, geographic locations are not considered in AURA.

Provisions have also been made for entering endogenously generated mission demand data. These provisions would be used if and when the resource management logic is expanded to permit endogenous decisions regarding mission demands. Such a decision would be communicated by calling the entry point SORTIE in the READFT subroutine where the mission would be entered into the mission demand pattern. If the unit is not specified, subroutine FRAG selects the unit best able to fill the demand.

2. STARTING THE MISSION

When the time specified for mission start occurs, subroutine MANAGE transfers control to subroutine FLIGHT. After checking that the mission need not be canceled because of weather conditions or attack damage, a check is made to see if vehicles have been assigned for a scheduled mission, or are available in the reserve force when the demand is unannounced. Each vehicle is checked to see if it has actually been readied for the mission. If crews are to be accounted for, subroutine FLYERS is called to locate a crew that is then tentatively assigned to the vehicles.

If fewer than the required number of vehicles are ready among those assigned to meet the specific demand, and if the demand has a priority at least equal to the minimum permissible level (as defined in Section IV), overwatch vehicles, later missions of the same or lower priority, and reserve forces of lower priority are each checked in turn for a

ready vehicle of the appropriate type and mission configuration. If, after all these sources are checked, the number of vehicles available to meet the demand is less than the minimum permissible number, the assigned vehicles and then the reserve vehicles are checked to see if vehicles are available that will be ready within whatever time is allowed for late "jump-off" or departure for vehicles of that type on that mission. If the minimum permissible number of vehicles have still not been located, the mission is canceled. If their number is sufficient, they are initiated with a call to subroutine LAUNCH that updates all the appropriate tallies and pointers.

Certain additional complexities will be noted in the FLIGHT and LAUNCH routines as a consequence of the options for composite missions and for late departures. When the minimum forces must be found for each of several different mission demands to prevent all from being canceled, it is necessary to withhold all departures until all missions have been checked. Furthermore, if, after checking several missions, it is found that at least one cannot be satisfied, it is necessary to modify various vehicle assignments and to release all tentatively assigned crews. Similarly, when a vehicle is going to depart late, it is necessary that certain data be retained until that time. To facilitate the latter operation the vehicle is placed in the vehicle delay heap until one AURA time unit after the vehicle's expected ready-to-go time; if it is still not ready to go at that time, the mission is canceled.

As each vehicle departs, it is checked for an abort; if one is to occur, the vehicle is scheduled to return to the unit with a full load of munitions after six minutes. It is handled like any other vehicle in the ensuing postmission inspection except that munitions are not required and attrition and battle damage are not assessed. If a vehicle is designated to recover at a different unit, that bookkeeping is also accomplished at the time that the vehicle is initiated. The actual departure is accomplished by placing the vehicle in the delay heap with the appropriate return time. The mission times for each vehicle in a mission are determined independently, unless recovery as a group has been specified on Card Type #16 for that type of vehicle and mission.

3. CREW MANAGEMENT

AURA's provisions for accounting for crews are controlled by the control variable CREWS; when initialized to 1, these features are activated.

Crew members are accounted for on an individual basis, much like vehicles. Each member is qualified for only one type of vehicle. Their assignments are managed so that each crew will receive a specified minimum amount of uninterrupted sleep during each 24 hour period and a specified minimum rest between missions. These two times are specified with the control variables SLEEP and REST. To avoid unnecessarily long shifts and early exhaustion it is presumed that crew assignments can be managed such that they remain off-duty until they are needed and will retire early whenever the demand permits.

Crews are managed with data that are stored in the PILOTS and PILOT arrays. PILOTS maintains a record of the number of crewmen in each unit, and pointers to the first and the last of those crew members who are on-duty and off-duty; these data are maintained separately for each vehicle type at each operating unit. The PILOT array maintains status information on individual crewmen, and pointers to the other crewmen with the same duty status.

The several operations required for crew management are carried out by different sections of the FLYERS subroutine. A crew is located for a tentative assignment by calling the entry point GETPLT, or SAVPLT for a late jump-off. When the vehicle begins a mission, the assignment is finalized with a call to the entry point FLYAC. When the mission has been completed, crew data are updated with a call to LANDAC. If the crew is due for a sleep period they are placed off-duty; if the vehicle was lost, but the crew was not, it is assumed that the crew cannot be reassigned for a minimum of four days.

In addition to these operations, entry point RELIEF is called at two hour intervals by MANAGE to check the on-duty crewmen and to relieve them as required. When the support area has been attacked, and the user has specified that a portion of the crewmen are lost, subroutine DISABL is called by subroutine BOMB to inflict the losses and update the crew information.

IX. SUPPORT AREA ATTACK AND RECOVERY

One serious disruption that an operating unit can experience is an artillery or air attack on their support location. Previous estimates of the damage likely to be sustained during such attacks on large installations (NATO airbases) and the lack of any generally agreed upon estimate of the real effects of such attacks on a unit's capabilities to recover and generate combat missions were motivations for the development of AURA's predecessor TSAR. And the highly irregular damage patterns experienced on locations that are subjected to conventional attack contributed importantly to the decision to create a model with sufficient detail that the critical effects of the highly stochastic damage patterns could be captured. Unless the possibilities for bottlenecks as well as for emergency and alternative procedures were included, one could hardly hope to represent the probable behavior of an operating unit during the crisis following an enemy attack.

1. SPECIFICATION OF ATTACK CHARACTERISTICS

In AURA, both operating and maintenance support units may be attacked and resources damaged or destroyed in accordance with the specifications supplied by the user on the basis of independent damage calculations. The user is free to schedule attacks at whatever times and locations he chooses, and AURA has been structured to accept fairly highly detailed damage data. These data may be entered at the beginning of the simulation and the scheduled attack times are placed into the heap in the ATTACK array; the various damage data are stored in compact form in the DAMAGE array.

The damage data supplied by the user for each attack may specify the percentage damage sustained by each type of the 11 classes of resources. For vehicles and facilities, the percentage damage sustained by the maintenance support personnel, support equipment, and the parts associated with the vehicle or facility at the time of the attack may also be specified independently for each of those resource classes. If the user prefers to simplify the input process for one or another class

of resource, he can omit the type specification, and all types of the specified resource class will sustain the same percentage loss.[1] This aid is available for all resource classes except crewmen, shelters, and facilities; for crews and shelters it is unnecessary, since they are all treated alike, and each facility or building must be designated specifically.

A special version of the AIDA (AIrbase Damage Assessment)[2] model has been developed to provide these data for AURA. Dubbed TSARINA (for TSAR INputs using Aida), this new computer model accepts detailed descriptions of the location, construction, and contents of various unit facilities, as well as detailed specifications of an enemy attack and weapons effectiveness factors, and converts the resultant Monte Carlo damage estimates into the format required by AURA.

TSARINA permits damage assessments of attacks on any complex composed of up to 500 individual targets (building, support equipment roads, etc.), and 1000 packets of resources. The targets may be grouped into 20 different vulnerability categories, and many different types of personnel, equipment, munitions, spare parts, accessories, and building materials can be distinguished. The attacks may involve as many as 50 weapon-delivery salvos (for air and artillery attacks) and 10 types of weapons. Both point-impact weapons (such as general-purpose bombs, artillery shells, and precision-guided munitions) and area weapons (such as napalm and cluster bomb units) can be accommodated.

TSARINA determines the actual impact points by Monte Carlo procedures--random selections from the appropriate error distributions. Weapons that impact within a specified distance of each target are classed as hits, and estimates of the damage to the structures and to the various classes of support resources are assessed using "cookie-cutter" weapon-effects approximations.

---

[1] Alternatively he can specify separately the expected damage to as many as 50 particular types of a resource class, and also separately specify the percentage loss of all other types of that class. This mixed option has specific requirements for the order the data are entered that are satisfied automatically if TSARINA has been used to generate the damage data.

[2] D. E. Emerson, *TSARINA: User's Guide to a Computer Model for Damage Assessment of Complex Airbase Targets*, The Rand Corporation, N-1460-AF, August 1980.

For each trial computation of an attack, TSARINA determines the fraction of each target covered by the circular damage patterns, and the results include estimates of the overall damage to each target and to all resource classes that are collocated with that target. In addition, the TSARINA output includes an estimate of the total percentage of each type of resource that was damaged at its various storage locations. These latter data are formatted to be loaded directly onto disk for immediate processing by AURA, or to be stored for subsequent use; no manual data conversion is required.

2. ESTIMATION OF THE STATUS OF RESOURCES AFTER THE ATTACK

When the time for an attack on a support area arrives, MANAGE transfers control to subroutine BOMB, which manages the subsequent operations until all of the specified damage has been dealt with, the shops and their activities reorganized as required, and the combat engineering resources allocated to high priority repairs. The stocks of the various damaged resources are decremented first. That process is straightforward for losses to off-duty personnel, munitions, accessories, building supplies, and the residual POL storage; it is somewhat more involved for on-duty personnel, crewmen, vehicles, shelters, and other facilities as will be described. The losses sustained by each type of each class of resource is computed separately. For all resources except vehicles and facilities (and the personnel and equipment actively engaged at the time of the attack) the losses are either determined as the expected value of the number lost or are sampled from the appropriate binomial distribution, depending upon the value of the control variable NONUNI.

Normally only off-duty support personnel losses are specified directly by the user; for on-duty personnel the basic AURA logic dictates that they suffer whatever losses would be expected when the facility to which they are assigned is struck or the vehicle they are working on is destroyed. The user is provided options, however, so that the casualty fractions for certain types of on-duty personnel can be specified independently of the status of the facilities. When crewmen are lost, they must be removed from the PILOT array and their pointer

system reorganized; this is accomplished with a call to the DISABL subroutine. For the vehicles and the facilities it is necessary to handle the damage to whatever other resources are present, as well as the damage to the resources themselves. For resources specified by the user with the #43 Card Types, orders are placed to replace the losses sustained with special shipments; such resources arrive following the specified delay for such shipments.

Vehicle shelters may be represented in AURA and a subset of these shelters may be designated for vehicles that are in reserve. If there are more vehicles at a unit than may be sheltered, the vehicles that are unsheltered are selected at random from among the nonreserve vehicles, and the remainder of the vehicles are assigned to a shelter with the reserve vehicles assigned first to the reserve shelters. A random number is drawn for each unprotected vehicle and compared with the likelihood that unprotected vehicles are damaged. Different damage probabilities may be considered for the alert shelters and for the other shelters. Each damaged vehicle is checked to see whether it is reparable, or whether it is suitable only for salvage. If the user has stipulated that personnel or equipment are lost, the on-vehicle tasks that were ongoing at the moment of the attack are each checked and the survival of the associated personnel and equipment is determined by comparing random numbers with the specified loss rates for these resources for each vehicle that was damaged in the attack. If resources associated with the task are lost, they are eliminated. If a vehicle is not reparable, it is next checked for parts that may be cannibalized for stock; for each part on the vehicle's parts list, a random number is compared with the specified parts loss rate to determine whether the part survived. If it has survived it is placed in the unit's stock of serviceables. The time to remove the parts is neglected on the assumption that that operation would be conducted as time permits. Only after these related resources have been checked are the vehicle records eliminated (using subroutines ENDAC and KILLAC). If the user has specified that lost vehicles are to be replaced with vehicles from CONUS or by filler vehicles that are held in reserve in the theater, subroutine ORDER is called to initiate that process.

When any of the other facilities (i.e., buildings) are damaged, another complex procedure must be followed. For these targets, the user-supplied damage data include the percentage of the facility that is damaged as well as the percentages of the maintenance personnel, support equipment, and spare parts that are lost.[3] The first step is to temporarily store these percentage damage data.

When all the damage data have been entered, control is transferred to subroutine REORGN, where the first steps are to define the status of resources present in the shop facilities damaged in the attack. The personnel and equipment that are considered to be at risk when a shop facility is hit are those engaged in parts repair jobs and those on duty at the shop and unassigned. If the user has designated that the activities of a particular shop are carried out at more than one location, the personnel and equipment that are at risk at each location are assumed to be in the same proportions as the user-specified job capacities at each previously undamaged location. Unless personnel and equipment have been assigned to separate companies or company teams, all on-duty unassigned personnel are assumed to be in the shop; if they have been assigned to a separate company or company team, the ready-line personnel are assumed to be in the facility numbered 30 plus the company number (e.g., Facility #32 for B Company). The unassigned on-duty personnel and equipment first are checked on a type-by-type basis and those assigned to a damaged facility are decremented appropriately. To check those engaged in parts repair jobs, the shops are checked individually; for each shop hit, the off-vehicle jobs are each checked and the associated resources reduced accordingly in the REPQ data. To maintain the parts records, it is necessary to distinguish the parts that were being repaired at the time of the attack and those that were waiting. For the parts under repair that are not lost, the jobs are placed in the interrupted task array. The serviceable parts present in the shop and the faulty parts not being repaired are then decremented. When the repair capacity of a particular shop is distributed among

[3] The user may exercise an option that disassociates the resource loss rates from facility damage by designating the overall loss rate for specific resource types; when this has been done, these specific rates override any resource loss rates entered with the facility damage data.

multiple locations, the reparable parts and faulty equipment that are being repaired or are waiting to be repaired at the time of the attack are assumed to be distributed among the undamaged locations in proportion to the capacities at the several locations. If all elements of a shop are damaged from a previous attack, the vulnerability of these resources are either zero or what they would be if the shops were undamaged (see variable ATRISK).

After the damage to the shop facilities has been processed, the surviving maintenance support personnel are reorganized using subroutine REDPEO. If some of the personnel have been assigned to ready-line units, the personnel of the same type in the several units are regrouped in the proportions implied by the "target levels" specified for each group of personnel; and then each group is divided into day and night shifts in the proportions implied by the "target" levels. This is done for all personnel types that suffered losses. Subroutine ADDAGE performs an equivalent reorganization for the support equipment that survive the attack. If the user has specified an amount of time by which vehicle maintenance activities can be expected to be disrupted, all tasks still in process on surviving vehicles, except for premission tasks, and in undamaged shops, are extended by that amount of time (i.e., SHPDLY). If any affected vehicle had been scheduled for a late jump-off, the vehicle and crew assignments are canceled.

Before the damage suffered by the facilities themselves can be dealt with, it is first necessary to estimate the current status of the facilities that had been damaged in previous attacks and that are being repaired at the time of the attack. It is assumed that the percentage of the original damage that has been repaired is equal to the percentage of the repair time that has passed. When the old damage has been updated and all combat engineering resources have temporarily been returned to stock, the present damage level is estimated. It is assumed that the damages due to the prior attacks and the current attack are independent and that they combine as $D = 1 - (1 - D_1) \times (1 - D_2)$, where D_1 and D_2 are the old and the new damage fractions.

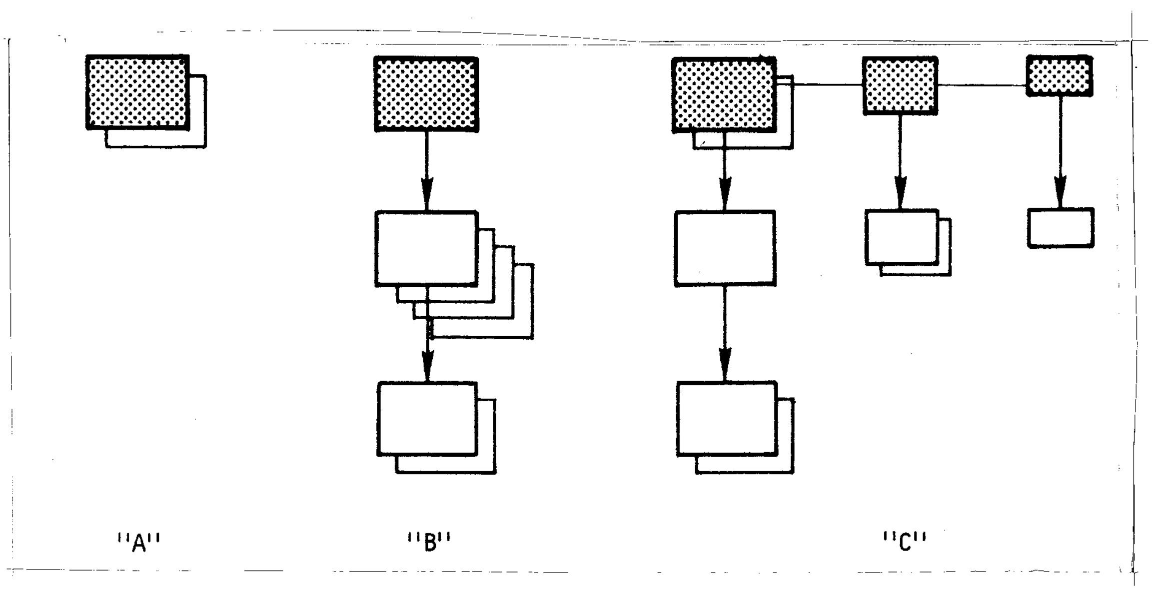
When the repair of a facility requires a sequence of procedures a check is made to see if any part of that sequence remains from a previous attack--i.e., if the facility is still undergoing repair. If

it is, the amount of damage (work) remaining for each step of the procedure is determined as noted above; that is, the residual damage at the time of the attack, and that imposed by the attack, are combined as though they were independent. Unless a specific damage level is entered for a step subsequent to the first, all steps are assumed to sustain the same percentage damage.

3. POSTATTACK RECOVERY AND RECONSTRUCTION

The actions of the combat engineers and other combat engineering resources in carrying out the emergency repairs essential for restoring critical functions can also be represented with AURA. As with any other feature of a computer simulation the representation falls short of capturing the complexities of the actual operations but nevertheless is thought to offer a considerable step forward in reflecting the grosser aspects of these tasks. Furthermore, the current formulation can be modified and improved as the community's understanding of how such a representation should be structured improves.

The optional representations of fixed maintenance facilities, and the procedures for restoring operations at those facilities are depicted below:



The shaded blocks constitute actual facilities as well as repair procedures. Thus "A" depicts the situation in which the activities of a given shop are carried out in a single location and damage to that facility can be repaired using either a basic repair procedure or--the backup blocks--one or another alternative procedure. The nature of the basic procedure is defined with the FACLTY array data for this shop, and the alternative procedures are defined by entries in the CERQTS (combat engineer requirements) array. The "B" depicts another shop whose activities also are carried in a single location, but three sequential civil engineering procedures are required to restore shop operations. Data defining each of these steps occupies a column in the FACLTY array.

Situation "C" is a distributed shop whose activities are carried out in three distinct locations, each of different size and capacity; the main location requires a three-step process to restore operations, whereas the auxiliary locations require only two steps. Each of the shaded locations must be defined in the CEPRTY array, and each of these locations and each of the subsequent procedures occupies a column of the FACLTY array.

The time for the repair or restoration of fixed facilities in AURA is related to the amount of damage, the type of structure involved, and the numbers of combat engineer personnel and equipment that can be brought to bear on the job. Each facility considered in AURA is distinguished by a facility number, a size, and the nature of its construction (or more correctly, the nature of the reconstruction or reconstitution required). The "facility" number is identical to the location of these descriptive data in the FACLTY array. The first 36 numbers are reserved for the reconstitution procedures to be used with the shops of the same number (and other special facilities); other locations in the array are to be used for data that describe alternative locations of distributed shops or subsequent steps in a repair process. Thus when more than one repair procedure is required to restore a particular facility to operational status, the descriptive data for the subsequent phases of the process are stored in otherwise unused columns of the FACLTY array. When the facility sustains damage, all steps of

such a repair sequence are assumed to sustain the same percentage damage, unless a specific damage level is entered for one or more of the subsequent repair procedures. Shops (and procedures) of like number will occupy "facilities" of the same number on the different units. Facilities #31, #32, and #33 are reserved as assembly points for unassigned maintenance personnel and support equipment (if it is an operating unit) when these resources have been assigned to separate companies or company teams. Position #34 is reserved for a special flag that signals the time that shop activities may be reinitiated. Locations #35 and #36 are not used in AURA. Locations #37 through #NOFAC are available for alternative shop locations or subsequent repair procedures.

When a building is damaged the "size" of the building and "percent damage" are combined to determine the magnitude of the restoration job. The requirements for the procedures used in repairing facilities of the differing types are filed in the CERQTS array. For each procedure some number of each of two types of combat engineer personnel and support equipment may be specified. The quantities specified in the basic procedure entered for each type of structure should represent the largest sized force that can reasonably be put to work on that type of job. For most types of jobs, alternative procedures should also be included (again in the CERQTS array) so that they may be adopted when insufficient resources are available. At this time, AURA does not consider the reconstruction of vehicle shelters (revetments).

The time and the quantities of the (up to) two types of building materials required for each facility repair procedure are specified in terms of the requirement for one "unit" of reconstruction; the magnitude of such a "unit" is defined by the metric the user chose in specifying the "size" of the facilities of that type.

In light of the possibilities for rather highly nonlinear relations between the repair time and the magnitude of the damage, the user is provided with 84 optional relationships that can be specified with a single number. The way these are used is as follows:

For each type of facility the user specifies the time required for a unit of reconstruction and a code number that defines the functional relation between total time and the magnitude of the task. If we define

t as the time for a unit of construction and N as the number of units of reconstruction that are required, the total time is:

$$T = \text{Delay}(B) + t \times N^{b}$$

and the code number that defines this relation is:

$$C = 12 \times P + (B - 1)$$

where b = g(P).

The function FTIME provides 12 choices for Delay(B) ranging from 0 to 48 hours (0, 1, 2, 3, 4, 6, 8, 12, 18, 24, 36, 48) and seven choices for b (g(P) = 0.5, 0.75, 0.9, 1.0, 1.1, 1.25, and 1.5). The code, C, that designates the functional form is interpreted in FTIME as:

$$P = \text{g.i.f. } (C/12) \text{ where g.i.f. is the greatest integer function}$$

and

$$B = C - (12 \times P) + 1$$

If, for example,

$$C = 48$$

then

$$P = 4$$

$$B = 1$$

and

$$T = tN$$

that is, a linear relationship between damage and repair time, since Delay(1) = 0 and b(4) = 1.0.

When subroutines REORGN and REORG2 have established the levels of the various resources that survived the attack, and the degree to which the various facilities have been damaged, control is passed to subroutine REBILD if the user has initialized the control variable CEWORK to one (and has provided the necessary data on the reconstruction

requirements). To facilitate the allocation of the combat engineering resources to repair the various damaged facilities, the user is also required to provide a priority listing of the order in which the facilities should receive attention; this list must include the locations of any distributed shops, whether or not they are to be repaired. The user also must indicate how many facilities on the list are especially critical, by initializing the control variable CRBLDG with the facility number of the lowest priority member in the critical range.

The first task carried out in subroutine REBILD is to check whether the combat engineer personnel and equipment are sufficient to initiate repairs of all damaged facilities in the critical range. If they are, subroutine INICON (initiate construction) is used to allocate the personnel and equipment and to withdraw sufficient building materials from stock to complete the job. And when the job completion time has been determined (as outlined above) these various task data are placed in the heap in the CEJOBQ array. This process continues until all damage is under repair or the resources are exhausted. To reflect the various disruptions that are not dealt with in this formulation but would delay the initiation of all reconstruction--for example, fires and roadway damage--the computed times are all increased by the value of the control variable CEDELY (combat engineering delay).

If the combat engineering resources are insufficient to start all the critical tasks, the allocation starts with the highest priority facility that is damaged and proceeds until resources are exhausted as was just described, except that the first alternative repair procedure is used when one has been identified.

When resources are exhausted, control is returned to subroutine REORGN, and when the user has included a GSRF in the simulation, one more thing must be done. For each shop at an operating unit that was damaged in the attack, a rough check is made to see if the parts could be shipped to and from the GSRF in less time than it is projected to repair the shop; if so, the faulty parts are shipped. This last task is carried out in the subroutine SHGSRF (ship to the GSRF).

4. COMPLETION OF COMBAT ENGINEER REPAIRS

When a combat engineer task has been completed, MANAGE transfers control to the entry point ENDCE in the subroutine BSEREP. A check is first made to see whether subsequent procedures are needed to complete the repair; if so, work is initiated as resources permit. If no additional work is required, the combat engineer personnel and equipment are released, the facility status is updated, and the released resources are assigned to the highest priority task that remains. When the repaired facility is a maintenance shop, the other basic task is to reinitiate the various shop activities, which is done by means of a call to the CHECK subroutine.

X. COMMUNICATIONS

AURA allows for the representation of scheduled shipments of material from CONUS to the theater, special shipments from CONUS in response to theater requests, intratheater shipments of resources, and the transmittal of combat unit status information. The schedules for each of these types of transfers are controlled by the user's specifications, as are the contents of scheduled CONUS shipments; the contents of the other transfers are generated endogenously.

1. SCHEDULED SHIPMENTS FROM CONUS

Resources scheduled to be delivered to the theater from outside the theater after the beginning of the simulation must be specified initially by the user. These data are entered with Card Type #31; the delivery times are arranged in a time-ordered queue in the CONUS array and the cards are stored in the CARGO array at the time of entry. The destination and time of delivery should be mentioned on the first of a set of cards when all the commodities on those cards are to arrive together.

The only resource classes that may not be shipped from CONUS are vehicle shelters and other facilities. No more than 99 units should be entered, except for munitions and accessories, for which the limit is 6250. If more are required, enter the commodity twice for the same delivery. For POL, AURA assumes that the unit of measure for shipments is thousands of gallons, whereas fuel normally is stored and used in ten gallon units. (Storage capacity for POL may be enhanced by specifying a shipment of Type #100 POL; units of measure are the same as for POL.)

When an arrival is noted in subroutine MANAGE, control is transferred to the RECSUP (receive supplies) entry point in the DOSHIP subroutine and the resources are added to the stock levels at the appropriate unit. When new maintenance personnel, support equipment, or vehicle parts arrive, subroutine CHECK is called to check whether they may be used immediately; for maintenance personnel, the new personnel are added to the day and night shifts to maintain the ratio of the shift

sizes in the same proportions as specified by the "target" levels for each personnel type in the initializing data for each unit.

When vehicles are moved to the theater from CONUS they are added to the inventory at the appropriate unit and undergo a normal postmission inspection, except that attrition is not checked. The crewmen are assumed to join the receiving unit and are given 24 hours to rest before their first combat mission. Crewmen that are moved to the theater (arrive without vehicles) are treated in the same manner.

2. RESPONSIVE SHIPMENTS FROM CONUS

The user also may simulate the requisition and resupply of resources from CONUS for any class of resources except shelters and the other facilities. When activated, a requisition is submitted for resources that are lost in combat or during an attack and for parts that can not be repaired in the theater. The resources requested are delivered after the delay specified by the user for each of the various resource classes. Arriving resources are treated in a manner identical to that described in the preceding subsection.

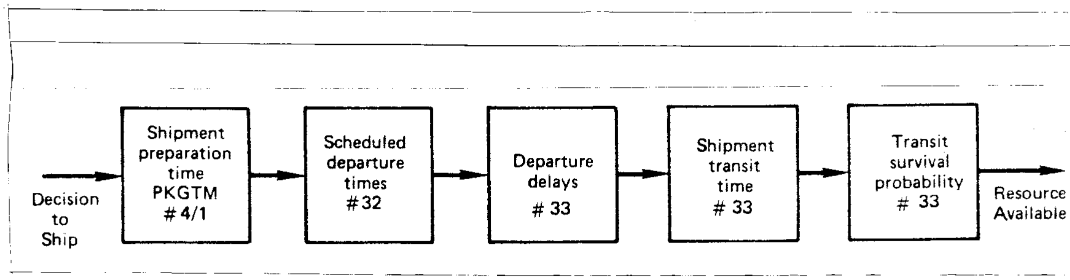
3. INTRATHEATER SHIPMENTS

Resources (except vehicles, crewmen, shelters, and facilities) may be transferred from one unit to another using an intratheater transportation system.[1] The description of this intratheater transportation system is controlled by the user's specifications of the schedules and the statistics governing their delays, cancellations, and losses. These shipments do not involve specific resources (e.g., trucks or aircraft) nor are they capacity limited; they only provide a representation of the times expended between the time that supplies are consigned for shipment and the time that a shipment reaches its destination and the contents added to unit supplies. The algorithms governing the transfer of resources with the intratheater transportation

---

[1] Vehicle and crew transfers can be affected exogenously by specification of a different final unit with a mission demand, e.g., as might occur when task forces are reorganized, or endogenously by directing vehicle transfer, as discussed in Section XI.1.

system are outlined in Section XI.2. The factors that are considered in this representation and the card types that are used to input the relevant data are summarized in the following sketch.



The user may specify daily departure times on an individual basis for each origin-destination combination. He may also control the mean departure delay, mean in-transit time, and the distribution of these values on an individul basis, using any of the 15 distributions that may be stored in the TTIME (true time) function. By manipulating the shape of these distributions a fraction of the shipments may even be canceled;[2] the commodities that had been prepared for that shipment are then scheduled on the next shipment. The user may also specify a loss rate for the shipments between any two units; the commodities on these shipments are not recovered.

The schedules for the various departures and arrivals are organized into time-ordered queues in the SHIP array with the subroutine SCSHIP (schedule shipments). These schedules are first organized at the time the program is initialized, and subsequently at midnight at whatever interval the user has specified with the control variable SHPFQ. Any of the schedules may be changed at any time during the simulation in much the same manner as the demands for vehicle missions (see the READFT subroutine for instructions).

The data stored for each shipment include pointers to the next departure from the same unit to the same destination, to the next departure from any unit, and to the next arrival at any location, as well as a pointer to the location of the first package to be included

[2] Delays greater than 18 hours are interpreted as cancellations.

with the shipment; the resources themselves are stored in the SHIPQ array.

Resources are prepared for intratheater shipment with a call to the SHPRES (ship resource) subroutine that checks on the availability of the quantity stipulated and decrements the shipper's stocks appropriately. For maintenance personnel, the work force is reorganized and reassigned as necessary with a call to subroutine REDPEO (reduce people). When the commodity is a faulty part rather than a serviceable part, that fact is denoted by a negative part number. When maintenance personnel, equipment, or serviceable parts are shipped, the numbers enroute to each unit are tallied in the appropriate storage arrays for these resources; faulty parts enroute to a GSRF, similarly, are tallied in that unit's portion of the PARTS array. The restrictions as to the size of the individual lots that are shipped are outlined in Section XVII of Vol. II. The quantities of these resources that are enroute are available for possible use in the theater resource management algorithms.

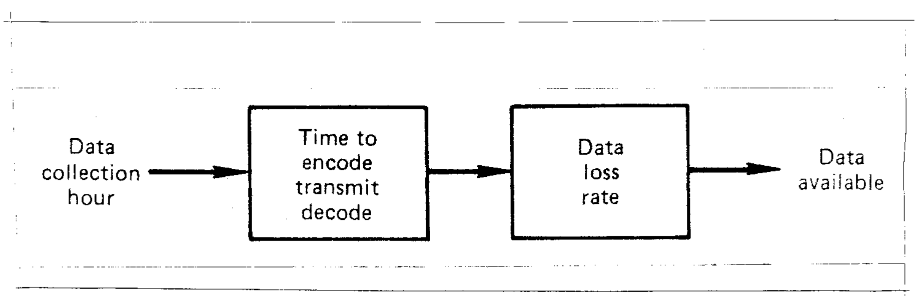
When an intratheater departure or arrival is noted in subroutine MANAGE, control is transferred to the subroutine DOSHIP. For departures it is necessary only to update the appropriate pointers; for arrivals, processing follows the same procedures outlined for the receipt of CONUS shipments, except that provisions are also made for the receipt of faulty parts and their transfer to the appropriate local repair shop.

As AURA is currently formulated, the intratheater shipment system is used only for parts, or for the shipment of maintenance personnel, support equipment, and serviceable parts when imbalances are noted among the operating units by the theater resource management system outlined in the next section.

4. INTRATHEATER RESOURCE STATUS REPORTS

Although an exact count is maintained on the status of all resources on all units throughout the simulation and these data could provide the basis on which various theater resource management systems might be examined, it seemed inappropriate to presume that the information that would be available to such managers would be precise and up to date. Indeed, one of the greatest drawbacks associated with

many centralized systems is their need for high quality communication and transportation systems. Unless some of the inefficiencies of the actual systems that may be available to our forces are represented, it is reasonable to question the validity of the results of any examination of schemes for managing resources on a corps or theater-wide basis. The following sketch indicates the factors that are considered:



All necessary data are entered using the #36 Type Cards.

In AURA, each unit may be designated to report the then current status of the maintenance personnel, support equipment, and vehicle spare parts at several different times during the day. These data are collected at those times for transmittal to "corps or theater headquarters"--to the corps or theater resource manager. The user also specifies the time delay before that information is formatted, transmitted, decoded, and available to that manager; the delays and the distribution of those delays are controlled unit by unit. Since only one location is provided for "in-transit" information, the data arrival time must be no later than the next data collection time; if it is, the transit time is shortened and a report of that action is printed. The completeness of the report may be controlled in two ways: the entire report may be lost in a specified percentage of cases, or some percentage of the individual data may not be reported.

When the user elects to activate the corps or theater communication system, the corps or theater manager's data unit is initialized with information that is accurate at zero time. However, if the user does not wish to turn control over to the corps or theater manager until some later time he may delay the initiation of the reports by initializing

OLDATA to unity. The main purpose of this feature is to avoid the related data processing during the early stages of a scenario, during which the user recognizes resource transfers would be unnecessary or undesirable. When OLDATA is first initialized to unity and subsequently set to zero by using the special code available in subroutine CONTRL, reporting will begin when the next report is due to be sent.

The particular status reports that are transmitted when the communication system is activated are controlled by the user's choice as to which classes of resources will be managed; he may select each or any combination of the maintenance personnel, support equipment, and spare parts, as explained in the next section.

Management of the corps or theater reporting system is the function of the STATUS subroutine. This subroutine is used first to place the various transmittal and receiving times into the REPORT array heap and to initialize the manager's data unit. Subsequently this subroutine is called by MANAGE whenever a report is to be transmitted or received. The data in transit and the data on hand for corps or theater management are stored together in the PEORPT, AGERPT, and PRTRPT arrays. The nature of this storage arrangement necessitates that reports in transit be received before the next is transmitted; it is the user's responsibility to assure that his schedules and delays make this possible.

When a report is to be received, a check is first made to see whether it was lost in transit. Subsequently a random number is drawn for each element of the data storage arrays and checked against the specified data incompletion rate. The last action in the STATUS subroutine is to store the time for the next day's transmittal or receipt of the corresponding daily report in the REPORT array heap.

XI. THEATER RESOURCE MANAGEMENT

AURA's ability to represent operations at a set of combat units and to handle the transfer of various classes of resources among those operating units and DS/GS maintenance units, other support units, and theater reserves can be combined to provide a unique mechanism for pretesting policies that would exert a broad span of control over corps or theater resources. Indeed, some may view AURA's prime role as a test bed for examining the effectiveness of new policy proposals. Although AURA's initial formulation is concerned only with the corps or theater-wide management of vehicles, maintenance personnel, and support equipment, in addition to faulty and serviceable vehicle spares, it could readily be extended to managing the other classes of resources.

The range of policy options that might be examined with AURA is limited, obviously, to those sets of decision rules that are expressible in terms of resource status information--past, present and projected--available within this simulation. In AURA there are basically three sets of status information available: (1) accurate data regarding current status, (2) the delayed and imperfect data provided with the corps or theater status reporting system, and (3) an approximate projection of each unit's current capability to generate missions. In addition, a limited amount of data exist regarding future mission demands, as well as the completion times for all ongoing tasks. The range of decisionmaking policy options that could be evaluated with AURA should be reasonably illustrated with the following discussions of those rules that are encoded in the current formulation.

AURA offers several options for managing vehicle resources. Initially, vehicles may fall into three different categories--those assigned to an operating unit, those in prepositioned war reserve materiel stocks (PWRMS) of "filler" vehicles, and war reserves materiel stocks (OWRMS) in CONUS. If the user designates a pool of "filler" vehicles, they may be used to offset degradations due to lost vehicles or to lost and damaged vehicles, as well as vehicles with excessive maintenance requirements and vehicles that have been withdrawn to a rear

unit for maintenance. These fillers may be used in addition to, or instead of, a reserve of vehicles in CONUS for replacing losses. The user is provided several options as to how these vehicles are used and managed, which are discussed fully in connection with Card Type #3/2 in Section XIX of Vol. II. When specified, replacement vehicles (OWRMS) are used exclusively to replace those lost in combat or from the effects of local attack, or have been so badly damaged they can only be salvaged for parts. If the user specifies that both fillers and replacement vehicles are available at the beginning of the simulation, the losses will be replaced with a filler and the replacement will join the filler pool when it arrives in the theater if it is not then needed at the operating unit.

AURA can automatically transfer vehicles from one unit to another based on an estimate of mission-generation capability, or if diversions of vehicles are necessary due to local conditions. This transfer occurs with a user-specified delay, and does not require a support resource except crew personnel. Because the movement of armored combat vehicles over long distances implied by such transfers would most likely be accomplished with the use of heavy equipment transporters (HETs) (unless the vehicle is an aircraft), and such HETs are not accounted for in the simulation, it is recommended that this feature of AURA not be used.[1]

All other theater resource management rules, or algorithms, are collected for convenience in subroutines CONTRL (control), CKGSRF (check GSRF), and REPRTY (repair priority). If it were desired to substantially expand these sections, additional subordinate routines could be appended readily. The algorithms now encoded deal with the following five resource allocation decisions:

1. disposition of serviceable spare parts repaired at an operating unit,
2. periodic review and reallocation of support personnel, equipment, and vehicle spares among the operating units,

---

[1] If the vehicle is an aircraft, then this subroutine can be activated.

3. acquisition of a spare part by an operating unit,
4. disposition of serviceable spares at a GSRF, and
5. choice among repairs waiting at a GSRF.

In several instances the user may select from alternative sets of rules for these kinds of decisions. The choices vary in both complexity and required processing; unfortunately there is a tendency for the more complex, often more efficient rules to require the greater amount of processing, and hence to absorb greater computer resources.

The sets of decision rules that apply for any particular simulation are dictated by the initial value of the control variables SHPREP and CMODE. When SHPREP is initialized, parts repaired at an operating unit are not automatically replaced in stock or sent to the unit where they were removed from a vehicle. Rather, a check is first made to see if the newly repaired part would reduce the number of deadlined vehicles at the unit where it was repaired, that is, whether (NORS Vehicle - Vehicle Missing Part) is less than SHPREP at that unit; if so, it is retained in the unit. If not, all units are checked and the part is sent to the unit with the greatest need. Newly repaired parts are always considered for shipment when SHPREP is large.

The user's choice for CMODE defines the three internal control variables CTHEA, CGSRF, and SHOPRY, since CMODE = 100 × CTHEA + 10 × CGSRF + SHOPRY. CTHEA controls the classes of resources that are periodically reviewed and reallocated; CGSRF controls the treatment of an operating unit's demand for a spare part and the disposition of newly repaired or newly acquired parts at a GSRF; SHOPRY controls the selection from among the parts waiting for repair at a GSRF.

The decisions and the units for those decisions that are controlled by these three variables are outlined below. Although each set of algorithms acts independently in the manner to be outlined, there are instances in which one rule may be overridden or negated by another. An obvious example occurs when a GSRF is directed to ship all newly repaired or newly acquired spares to one of the operating units *and* all operating units are directed to order a part from central supply at the

GSRF; such requests will always go unfilled if all parts have been shipped as soon as they become available. The user must be aware of the effect of his choices for the various control variables and of their possible interactions.

1. MANAGEMENT OF FILLERS

AURA options for managing vehicle resources are designed to simulate various decisions that corps or theater managers would, in certain circumstances, make to enhance the mission-generation potential of their combat force. Included would be the replacement of lost vehicles, the insertion of reserve vehicles to offset vehicles immobilized by the need for extended maintenance, and various work-leveling decisions. Such situations could arise whenever units suffer disproportionate losses of support resources or vehicles, or when local disruptions force vehicles to divert.

A pool of filler vehicles may be defined at the corps or theater level and used to offset the degradations due to lost or damaged vehicles, as well as those with excessive maintenance requirements. This pool may be used in addition to, or instead of, a reserve of vehicles in CONUS. It is assumed that a crew is available for each vehicle in the pool. The control variables FILLAC, MAXMNT, and FLEVEL provide options as to how these vehicles are used and managed.

The value for FILLAC defines the circumstances under which a filler vehicle is assigned to an operating unit. The five conditions are:

FILLAC	Conditions for Filler Usage
1.	A vehicle is lost during an operation.
2.	A vehicle is lost, or is transferred to a rear DS/GS maintenance unit for battle damage repair.
3.	A vehicle is lost, or is transferred to a rear DS/GS maintenance unit for maintenance, including battle damage repair.
4.	As in 2, or when the expected repair time for a battle damaged vehicle exceeds MAXMNT, and the FLEVEL conditions are met (see below).

5. As in 3, or when the expected maintenance time for a vehicle exceeds MAXMNT and the FLEVEL conditions are met.

Whenever a filler vehicle is assigned to a combat unit to replace a combat loss, a replacement is ordered from CONUS, if stipulated by the replacement policies prescribed with the Type #43 Cards.

The value of FLEVEL affects the decision to augment individual unit vehicles, and controls the disposition of both vehicles repaired at a rear DS/GS maintenance unit and those transferred from CONUS to the filler pool. In AURA, it is necessary that the user indicate which condition he would like to hold in order for the requisition of a filler vehicle, or the return of one from the rear to proceed. These conditions are:

FLEVEL	
0	The number of surviving vehicles is less than the number authorized to the unit.
1	The number of non-battle-damaged surviving vehicles is less than the number authorized to the unit.
2	The number of surviving vehicles is less than the unit's shelter capacity.
3	The number of non-battle-damaged surviving vehicles is less than the unit's shelter capacity.

When these conditions are not met, vehicles newly repaired at a rear DS/GS maintenance unit and vehicles that have arrived from CONUS are consigned to the pool of filler vehicles.

2. PERIODIC REVIEW AND REALLOCATION OF RESOURCES

The available numbers of support personnel, equipment, and serviceable vehicle parts may be reviewed periodically, and actions taken to redress serious imbalances that are noted. The nature and timing of these reviews are controlled by CTHEA and the user's choice for C4TM and C4INT. The first review occurs at the C4TM'th hour of the simulation and subsequent reviews are at intervals of C4INT hours. The delayed and imperfect status data reported to the corps or theater manager by the theater communications system are used in these reviews.

The particular classes of resources reviewed at those times are dictated by the value of CTHEA as follows:

PERIODIC THEATER-WIDE RESOURCE CHECKS

CTHEA	PERSONNEL	EQUIPMENT	PARTS
0	-	-	-
1	-	-	X
2	-	X	X
3	X	-	X
4	X	X	X
5	X	X	-
6	-	X	-
7	X	-	-

In AURA, it is recommended that CTHEA be set to zero. Only special analyses might require an alternative setting.

3. ACQUISITION OF SPARE PARTS

Whenever a vehicle "hole" is reported, that vehicle's operating unit may, under certain conditions, request and, if other conditions are fulfilled, obtain a spare part from another operating unit or from the theater's central supply. The procedures used are controlled by the value of CGSRF, which also controls the rules governing the disposition of newly repaired and newly acquired parts at the GSRF. The procedures adopted are as follows:

CGSRF	UNIT REQUESTS FOR PARTS	GSRF DISPOSAL POLICY
0	No response	Return to sender
1	Filled by first unit fulfilling conditions	Return to sender
2	Filled by unit best fulfilling conditions	Return to sender
3	Filled by GSRF when conditions permit; otherwise same as 1	Retained in stock
4	Filled by GSRF when conditions permit; otherwise same as 2	Retained in stock
5	Same as 3	Send to most needy unit if in excess of req'd reserve
6	Same as 4	Same as 5
7	Filled by GSRF when conditions permit; otherwise GSRF directs lateral resupply	Same as 5

The procedures and conditions that govern the five different responses to a unit request follow:

a. When CGSRF = 1, a simple mode of lateral resupply is simulated. Whenever a "hole" is reported, the nearby units that the user has specified (a maximum of four using the special version of Card Type #23) are checked one by one in numerical order, and the first unit that fills the specified conditions ships a part to the requesting unit. Those conditions are, first, that the number of reparables minus the number of "holes" at the requesting unit is less than the value of ORDER2, and, second, that either (i) the donating unit has at least two serviceable parts, or (ii) the donating unit's adjusted stock requirement--i.e., (Nominal Stock Level) × (Current Number of Vehicles) ÷ (Nominal Number of Vehicles)--is less than one-quarter of a part. As the value of

ORDER2 varies from a positive integer to zero to a negative integer, the policy for requesting lateral resupply can be varied from very liberal to very strict.

b. When CGSRF = 2, the procedures parallel those for CGSRF = 1, except that all units are checked and the unit with the largest number of serviceable parts is chosen; if the donating unit has only one serviceable part, the current value of its nominal stock level must again be less than one-quarter of a part.

c. For values of CGSRF greater than 2, the first action taken by the ordering unit is to check whether the theater's central supply has a part that may be shipped. If there is a serviceable part at the central supply point, it is shipped if the requesting unit fulfills the following condition: the sum of the ordering unit's number of reparables, plus the number of serviceables already enroute from the central supply, minus the number of "holes" in vehicles at that unit must be less than the value of ORDER1. Again, a negative value of ORDER1 defines a strict lateral resupply policy, under which parts can be requested only when the number of outstanding "holes" exceeds the tangible assets by the specified (negative) level.

If a part is not shipped by the GSRF, the requesting unit then attempts to obtain a part from an operating unit by a lateral resupply action. For CGSRF = 3 and 5, the same procedure is used as when CGSRF = 1. For CGSRF = 4 and 6, the procedure is that used when CGSRF = 2.

d. When a part cannot be shipped by the GSRF, and CGSRF = 7, the central manager checks the other operating units to determine which can best afford to ship a part to the requesting unit. This check of the other units is based on the status information as reported through the theater reporting system. To select the donor unit the following ratio is computed for all other units: (available parts plus enroute parts) divided by (the current level of the nominal unit requirement). The unit with the largest value for this ratio is directed to ship a part to the requesting unit, if that value is greater than one-quarter. If it is not, but there are at least two serviceable parts at that unit, one is shipped.

4. DISPOSITION OF NEWLY REPAIRED OR NEWLY ACQUIRED PARTS AT A CENTRAL SUPPLY POINT

As outlined at the beginning of the preceding subsection, three options are available for disposing of newly acquired serviceable parts at the corps or theater control supply point. For CGSRF = 0, 1, and 2, newly repaired parts are returned to the unit where the reparable was generated; newly acquired parts are placed into the local stock. For CGSRF = 3 and 4 all such serviceables are placed into stock at the GSRF.

For CGSRF = 5, 6, and 7, any newly acquired part that is in excess of the central supply's stipulated reserve is shipped to the most needy unit. That determination is made in the same manner outlined in conjunction with periodic resource reallocations; that is, it is sent to that unit with the lowest ratio of (serviceables + reparables + enroute - "holes") divided by the units' current nominal requirement. These calculations are based upon the status information reported by the corps or theater reporting system.

5. REPAIR PRIORITY DETERMINATION AT A GSRF

When broken parts must wait to be repaired at an operating unit, their position in the appropriate shop's wait queue is based upon the local supply and demand, when the control variable ORDWT has been initialized as unity, as outlined in Section VII. At a centralized repair facility somewhat different procedures naturally must be followed when ORDWT = 1, since there is no local demand, as such. Interrupted repairs are given priority over waiting repairs when resources become available, on the assumption that if they were sufficiently important to have been started, they should be finished. But when resources are available and no interrupted repairs are queued, the parts that have been waiting are checked to see which should receive attention. Actually, the prioritization of waiting parts at a GSRF is a two-step process; one set of rules governs the order in which parts are placed into the wait queue, and the second governs the criteria that must be satisfied when a part is withdrawn from the queue. The primary purpose of the first of these two procedures is to limit the processing required for carrying out the second procedure.

Whenever a reparable completes the administrative delay at a GSRF and must wait to be repaired, it is ordered in the wait queue by ascending values of the following quantity (i.e., items with low values receive priority over those with high values) when ORDWT = 1:

(Repair time)/[(Vehicles with holes)(Relative importance)]

Thus parts that are important, needed, and can be repaired quickly receive priority. This parameter takes into account all vehicle types that use that particular type of part, and the importance[2] of that part to the missions that those types of vehicles can accomplish. If ORDWT is zero, the items are ordered FIFO (i.e., first-in first-out).

When maintenance personnel and/or support equipment are released at the completion of another repair job, and when ORDWT = 1, two alternative sets of rules may be used for selecting the next part that will be repaired. The choice is made in subroutine CKGSRF, which is called whenever resources are released; the choice is controlled by the variable SHOPRY (shop priority). For either of the options, the demand outstanding for the first item in the queue is empirically estimated, and, if the value of that estimate is greater than the threshold variable INDEX, the repair is initiated without checking any further. If it is not, the next part is checked, etc. If the value for none of the parts exceeds the threshold, the one with the highest value is initiated.

The manner in which the demand outstanding for a given part is estimated is controlled by SHOPRY. When SHOPRY = 1 the demand is estimated as the product of the number of vehicles that require the part, times the part's importance, where "importance" is as defined above.

When SHOPRY = 2, the estimate of demand takes into account both the

---

[2] The PRTCRT array is generated during initialization: For each part, each entry in this array contains, in packed form, a record of which vehicle types use that type of part, and which of that vehicle's missions require that part. The relative importance of a particular part, as used above, is defined as the sum of (number of mission types for which the part is required) divided by (number of mission types that can be accomplished) for the several types of vehicles using the part.

current backlog of repairs and the expected future demands for parts based on the present pattern of mission demands. Demand is expressed as proportional to the sum of (1) a fraction of the existing number of "holes," and (2) the number of "critical holes" that would be expected to develop over the average shipping time if the current mission demands continued and were met. The fraction of the existing "holes" included in this calculation is:

F = (the current number of missions demanded for which the part is essential)/(the current number of missions demanded of vehicles that use the part)

which is determined by summing the demand for missions across the various types of missions and vehicles. The expected number of critical holes that would be generated is estimated as

E = (the current number of missions demanded for which the part is essential)/(the mean number of missions before failure of the part)

Thus the overall demand, or relative importance, of repairing any particular part is taken to be

$$\text{DEMAND} = 10 \times [\, F \times (\text{ existing "holes" }) + S \times E \,]$$

where S is average shipment time in days, and the factor 10 has been introduced simply to maintain distinctions among the integer values of the demands. As with the other option, repairs are initiated for any part whose "demand" exceeds the value of the variable INDEX; if no value is that large, repairs are initiated on the part with the largest value of demand.

XII. DATA INPUT

The first step of the input process is to zero out all storage arrays and define their dimensions; this is the primary function of subroutine INIT. That subroutine also contains a variety of material that will assist the programmer in accomplishing whatever redimensioning is required to tailor AURA to his special requirements. These materials include the 20 primary sets of named COMMON, a complete list of the arrays that are found in COMMON, data clarifying which array dimensions may be modified, and extensive comments to explain how such changes should be made. Many of these materials may also be found in Vol. II of the User's Manual. Whenever AURA's array structure is redimensioned, special care should be taken to assure that the loops used in subroutine INIT to zero out the corresponding space span the correct range.

The second step in the input process is to read the input data provided by Card Types #1 through #49 using subroutine INPUT, INPUTA, INPUTB, and INPUTC. The definitions, formats, and procedures for entering these data are outlined at length in Section XIX in Vol. II.

This process has several built-in checks (actuated when VERIFY > 0), but the user should adhere precisely to the instructions; when VERIFY = 3, each input card is screened by subroutine TESTER, which has been designed to catch a variety of common errors. The user has considerable latitude as to what is to be included; many portions of AURA may be inactivated simply by omitting a card or by providing a null entry for certain data.

Subroutine INPUT calls on subroutine INPUTC to read attack data and maintenance unit damage data and to organize the attack times in a heap. The INPUTC subroutine is designed so that these data may either be input directly with the AURA data deck or read from disk, where they have been stored by the companion model TSARINA, which computes the required damage data from a description of the attacks and the location of resources among the various maintenance facilities.

In addition to simply storing data, subroutine INPUT, assisted by subroutine WRAPUP, also arranges resource shipments from CONUS in a time-ordered queue, computes the entries for the SHPTSK (shop tasks) array, and uses subroutine AVGTME (average times) to compute the average time that each work center (shop) can be expected to spend on on-vehicle tasks and off-vehicle repair jobs for each type of vehicle, when unit resources are unlimited. These estimates take into account the likelihood that the different tasks will arise, parts will be broken, and the parts will not be thrown away or shipped to another location for repair.

In addition, subroutine WRAPUP uses subroutine RREQTS to compute the expected requirements for maintenance personnel, support equipment, parts, munitions, and shop facilities (vans) for each type of mission and each type of vehicle when the control variable STATE has been initialized to a value greater than zero. These estimates are used subsequently in subroutine BASCAP to provide daily projections of each unit's mission-generation capabilities.

When the user has elected to let AURA initialize the parts data and the parts pipeline to the depot, as outlined in Section VII.1, subroutine COMPRT (compute parts) is called next by WRAPUP. When this option is chosen the user must first have stipulated certain unit characteristics and the NRTS policies for each part and each type of unit (using special versions of the Card Type #23). Subroutine COMPRT manages subroutine IPARTS and IPART2, which compute the total numbers of each type of part for each type of unit. Listings of the results of these computations and of the pipeline contents are controlled by PPRINT.

The user has two options for recording the input data: simply to list input data as they are entered, the user places a "1" in column N of the first input card, if Card Type #N is to be listed. The first input card is an unnumbered Card Type that appears just before the Type #1 Card and is called the "Card Image Print Control Card." The other option lists the data after they are stored and after the various special initialization actions have been carried out. This option is requested with the special card that precedes the mission demand data;

again a "1" is placed in column N if the data read with Card Type #N are to be reproduced. This unnumbered card type appears just before the first Type #50 Card, and is called the "Formatted Input Data Print Control Card."[1] The subroutine INLIST and the support routines LIST1, LIST2, LIST3, LIST4, and LIST5 respond to these demands. The user should note that the data are printed directly from storage and that they frequently have been modified or "packed" differently than when they were input.

The last steps in the input procedure are managed with subroutines INITIZ and TRIALS. The pointers identifying the available space in the several dynamic storage arrays are initialized in INITIZ. The last step in subroutine INITIZ is to list the status of personnel substitutability at each combat unit. Initialization is completed in subroutine TRIALS; when the control variable STATE has been initialized to a value greater than zero, subroutine BASCAP is called to generate the initial projection of each unit's mission-generation capabilities. These approximate mission projections are derived by comparing the average resource demands for each type of mission and each type of vehicle with the available quantities of those resources at each unit, as outlined in Section IX.1. These projections are subsequently updated each evening at 1930 and are used with a variety of algorithms concerned with managing vehicle assignments and transferring vehicles from one unit to another.

If theater resource reports are to be transmitted during the simulation, TRIALS next calls subroutine STATUS to initialize the corps or theater manager's data base with up-to-date and complete information regarding the resources that will be managed. The intratheater shipping schedule queue is organized next.

The next step in TRIALS is to input the initial set of mission demand data by calling the entry point DAYONE in subroutine READFT (read mission data). As explained at greater length in Section VIII, these data can be replaced or modified each day at 2000 or, if the missions are periodic, they may be used to control the demand for missions for several days or throughout the simulation. Finally, the heap in the

---

[1] To get a visual idea of how these cards appear in an AURA input deck, the user is referred to Fig. 1 of Vol. II.

array PERIOD (periodic and scheduled tasks) is initialized in subroutine MANAGE (using entry point MANAG).

The input procedure up to this point has been primarily concerned with acquiring the data that describe the various tasks and the initial resource levels and schedules, and with initializing various queues and heaps. The initial status of the vehicles and the maintenance shops is established with Card Types #41 and #42, and, when parts are initialized automatically, deadlined vehicles may have to have been designated to provide sufficient parts to stock the pipelines, should the unit be understocked before operations commence. If only Card Type #41 is used, it is presumed that in the situation being simulated, there has been an opportunity to work off all unscheduled maintenance tasks (except for those vehicles so designated above), and to load the vehicles for some type of mission at the beginning of the simulation. Similarly, the parts stockage generation option presumes that all spare parts are serviceable. Thus the various shops are inactive and no jobs have been interrupted or are waiting. To be consistent, the available maintenance personnel and support equipment should have been set to their maximum values.

To reflect a situation in which vehicle maintenance tasks remain to be completed and various parts are being repaired or are waiting to be repaired, subroutine ZSHOPS is called by subroutine TRIALS. This modification of the initial conditions is controlled by Card Type #42, where in-process vehicle maintenance is expressed by a three-part distribution for each type of vehicle at each unit. Thus one might specify that 20 percent of Vehicle Type #2 at Unit #3 has two tasks outstanding, 30 percent have three tasks, and 10 percent have five tasks. Subroutine ZSHOPS selects the required tasks at random, consistent with their nominal probability of occurrence, and computes the time remaining as a random fraction of the normal task time. If parts repair jobs are specified on Card Type #42, the appropriate numbers are selected and placed in the administrative delay queue or in an in-process status by an equivalent random process.

The default crew status (established in subroutine INPUTA) is that all crewmen will be available for assignment at any time after 2:00 AM on the first day. If some crewmen must receive more (or less) rest, the

appropriate elements of PILOT(3,I) would need to be reset to the proper time (in AURA time units).

When all phases of the initialization process are completed, program execution is terminated if VERIFY was set to 2 or more, either by the user or in response to input errors that AURA detected.

XIII. SIMULATION CONTROL

The MAIN routine initiates and concludes the simulation but delegates the control of the three main phases--input, simulation, and output--to subordinate routines. Input has been discussed, and printout of the final results is controlled by subroutine OUTPUT. Control for the simulation proper is passed first to subroutine TRIALS, which is responsible for the last portions of the initialization process and for running the simulation the designated number of times. TRIALS manages the storage of the initial conditions for the first trial, for regenerating zero-time shop activities, and, when spares stocks are computed internally, for recomputing the initial spares for each trial. Control is passed by TRIALS to subroutine MANAGE, which exercises primary control throughout each trial of the simulation.

The basic task performed by subroutine MANAGE is to examine the earliest event that will occur in each of eight separate groups of events and to determine which of these eight is to occur next. Simulated time is then advanced to that time, and control is transferred to the appropriate subroutine for processing that event.

If the next event in each of two or more of these eight groups of events is to occur at the same time, the first event examined is processed first. The order in which the groups are examined is:

1. Completion of combat engineer reconstruction jobs
2. Completion of on-vehicle maintenance tasks
3. Completion of off-vehicle parts repair jobs
4. Periodic and user-scheduled events
5. Completion of vehicle delays (dead times)
6. Mission start events
7. Completion of munitions assembly jobs
8. Arrival of resupply shipments

This order has been established primarily with a view toward minimizing unnecessary processing. Thus shop reconstruction is checked before maintenance personnel are released, so that parts repairs that are awaiting initiation would not need to be checked twice. On-vehicle tasks and vehicle delays are completed before missions are checked, so that vehicles that are becoming available for dispatch are so designated at departure time.

Control is transferred to subroutine BSEREP, RUNAC, RUNSHP, FLIGHT, DOBILD, and DOSHIP for the first, second, third, sixth, seventh, and eighth groups, respectively. For the fourth and fifth groups the nature of the event or delay determines which subroutine takes control; subroutine MANAGE transfers control to the appropriate location.

Many of the user-defined management control variables may be changed during the simulation. The time at which any such change is to occur may be specified in the input data, or may be selected endogenously, thus providing a form of dynamic adaptive control. Subroutines ADAPT and MODIFY provide for the management of the user-supplied logic that controls such adaptive behavior.

When processing has been completed by the subordinate routine(s), control is returned to subroutine MANAGE and the next earliest event is selected. The entire simulation proceeds in this manner until the user-stipulated simulation length is exceeded, at which time MANAGE returns control to TRIALS to print the trial results and to initiate the next trial, or to print the overall results of the several trials.

XIV. SUPPORT SERVICES

Fifteen subroutines and 11 minor subroutines and functions support the main simulation. Each performs one or more specific functions and many are called upon from a variety of different locations. The functioning of each of these support routines is described at least briefly in this section. These discussions are ordered alphabetically; the subroutines discussed include:

BREAK	Computes break-rate modifiers for on-vehicle tasks when VBREAK is unity
CHECK	Checks on outstanding demands for newly available resources that have been released from vehicle maintenance tasks and parts repair jobs
ENDAC	Eliminates all records associated with a vehicle
FTIME	Generates time requirements for combat engineer jobs
HEAP	Enters and removes items from a heap
INTRUP	Enters and removes time-ordered items from the interrupted task array
KILLAC	Eliminates all records for a vehicle that is lost in combat
LOSSES	Determines the specific number of items lost
NORRPT	Enters and removes records of vehicle "holes" from the NORQ array
REDPEO	Reduces or increases the number of support personnel and reorganizes the shift structure
RESET	Resets simulation time for a continuous simulation of NTRIAL $\times$ SIMLTH days duration when EXTEND = 1
SHIFT	Adjusts the size and activity of the work force when shifts are changed
STRTSK	Stores and retrieves required and deferred on-vehicle tasks
TTIME	Generates time requirements for vehicle maintenance and theater communication delays
WAIT	Enters and removes time-ordered items from the waiting-task array

The other 11 service items are summarized briefly at the end of this section.

Subroutine BREAK

This subroutine is used when VBREAK is initialized to unity to modify the probabilities with which on-vehicle tasks are required as a function of achieved mission rate. Called at the end of each day, this subroutine computes the mission rate achieved during the preceding day for each type of vehicle; the estimate is given by the total missions achieved by each vehicle type and the total number of such vehicles surviving in the corps or theater. The appropriate break-rate modifier is then computed separately for each shop and each vehicle type on the assumption that the nominal break-rate applies for a mission rate of one per day, and that the actual break-rate falls linearly for each additional mission per day by the percentage specified with Card Type #44. The resultant value is stored in the second element of the TSKPR array.

Subroutine CHECK

This subroutine is used to check on resource demands that may be outstanding and is called whenever resources are released from a previous event or are delivered to a unit. The five sources for such demands are interrupted, waiting, and deferred on-vehicle maintenance tasks and interrupted and waiting parts repair jobs. To reduce processing somewhat, the call to this subroutine may specify a shop number, a vehicle, a part, a type of personnel, a type of equipment, or any combination. In the subsequent search among outstanding demands no attempt is made to initiate tasks that do not require the resource specified.

The search is ordered so as to examine on-vehicle tasks before parts repair jobs; in each case, interrupted items are examined before those that are waiting. At night (after ENDAY), deferred tasks are checked after repairs have been checked. All five queues are searched when a shop or a type of support personnel is specified. If an equipment type is specified, only on-vehicle tasks that are waiting (and, at night, deferred) are checked. When a vehicle is specified, only on-vehicle tasks are examined. When an LRU is specified, all

vehicles waiting for that part are examined to select the one with the fewest holes, and if two or more have the same number of holes, the vehicles with the earliest projected ready-to-go time is selected. When the part specified is an LRU, only jobs waiting for repair are examined.

For the shops that may lend maintenance personnel or support equipment to other shops, subroutine CHECK next checks the shops that are listed in an AURA-generated list of borrowing shops to see whether the newly released personnel or equipment are needed either for on-vehicle or for off-vehicle jobs. If so, the resource is lent to the shop that is found to require it.

Subroutine ENDAC

This subroutine is used only when a vehicle has been damaged or destroyed by an enemy attack. It is called from subroutine BOMB after that subroutine has appropriately decremented the maintenance personnel, support equipment, and parts associated with the vehicle at the time of the attack.

For damaged vehicles ENDAC is used to eliminate any mission assignments; if a vehicle has been destroyed, ENDAC removes it from the delay queue (when appropriate), and removes all references to required, interrupted, or waiting tasks. All ongoing tasks are then stopped and the surviving personnel and equipment are released for other jobs. No times are recorded for these tasks. The last step is to call subroutine KILLAC to erase any deferred task records, to eliminate the vehicle from the unit inventory, and to order a replacement vehicle, as appropriate.

Function FTIME

This special function provides the user substantial flexibility for specifying how the required times for combat engineer jobs vary for different types of jobs and different levels of damage. In basic terms, the formulation consists of a delay, or start-up time, plus a damage-dependent reconstruction time. For each type of combat engineer job, the user specifies the time (t) required to repair a "unit-of-damage" and indicates how the total time (T) will vary with the total number of

units of damage (D) by entering a coded number (C) for the functional relationship. This subroutine uses those data to estimate total time as follows:

$$T = \text{Delay} + t \times D^b$$

where

$$\text{Delay} = f(B)$$
$$b = g(P)$$

and, since $C = 12 \times P + (B - 1)$,

P = g.i.f. $(C/12)$ where g.i.f. is the greatest integer function;
$B = C - (12 \times P) + 1$.

The data tabled in FTIME for f and g provide 12 values for the delay (0,1,2,3,4,6,8,12,18,24,36, and 48 hours) and 7 values for b (.5,.75,.9, 1.0,1.1,1.25, and 1.5). To specify a time proportional to the total damage, without any initial delay, C would be 48--i.e., P = 4, B = 1, so that b = 1.0 and Delay = 0.

Subroutine HEAP

When it is not necessary that timed events be ordered, but only that the earliest event be located readily, a data collection that has been called a "heap" permits more efficient processing. On the average, only two positions need be checked when a new event is entered into a heap.

This subroutine has four entry points, one to enter an item (INHEAP), one to remove the item with the lowest valued time (OUTHEP), a third to extract an item (EXHEAP) from within the heap, and a fourth to modify the time for an item in the heap (MODHEP). To extract an item from the heap, or to modify an item, it is necessary to know which column it occupies in the parent array, if it is to be found readily; but when these actions are required before an event has become the one with the earliest time, that information logically is known.

The size of the calling array is a variable and the number of entries in that array may be fixed or variable. This subroutine operates on three rows of whatever storage array is specified in the

calling statement. The time of the event is in the first row, a pointer to the event's position in the heap is in the second, and a pointer back to the event from its position in the heap is in the third row.

One peculiar property of this data structure should be noted: If several events are entered that have the identical time associated with each of them, they will not be removed in the same order in which they were entered.

Subroutine INTRUP

On-vehicle tasks and parts repair jobs that have been interrupted are queued in the INTTSK array. Each shop at each unit stores a pointer to the first and the last of its interrupted tasks and its interrupted repairs in the array SHOPS. Whenever resources are available to start an interrupted activity, the first item in these queues is the first to be examined. If the user wishes priority to be given to the item that has been in the queue the longest, the control variable ORDIT is initialized to zero and the queue is managed locally in the main routines.

If the user wishes to have the events ordered such that the one with the lowest value of a variable called TIME is first, ORDIT is initialized to unity, and subroutine INTRUP is called to manage the queue. The value of the variable TIME need not be a time, per se, and, as discussed elsewhere, differing events are queued in accordance with differing definitions for TIME.

This subroutine has separate entry points for entering an item (ININT) and for removing an item (OUTINT). The code is written so that any item may be removed, not only the one that is first in the queue. Three rows of the INTTSK array are involved in queue management; the value of the variable TIME is stored in the seventh row, and pointers to later and to earlier items are in the fifth and sixth rows respectively.

Subroutine KILLAC

This subroutine is used whenever a vehicle is lost in combat, and it also completes the work begun in ENDAC when a vehicle is destroyed at the support location. The two basic functions performed by this subroutine are to erase any reference to vehicle tasks that may have been deferred and to eliminate the vehicle from the unit inventory.

Subroutine LOSSES

This subroutine generates the specific number of items that are lost when N items suffer a loss probability of P. If the control variable NONUNI is zero, the value returned for one item is determined by comparing a random number with P; for more than one item the value returned is that integer closest to the expected losses--i.e., N × P. If the control variable NONUNI is unity, the value returned is a sample drawn from the binomial distribution (determined by comparing N random numbers with the value P).

Subroutine NORRPT

Whenever a part has been removed from a vehicle and has not been replaced immediately, or whenever a part on a vehicle has been found to be defective but has not yet been removed, a record is made of the particular vehicle and part. The data on these reports, or "holes," are stored in array NORQ using subroutine NORRPT.

Whenever an entry is made in NORQ (using entry RPTNOR) this subroutine first adjusts the number of vehicles in the unit that are missing a part and then adjusts the count of the "holes" in that vehicle. If the rules for intratheater resource transfer permit, an order may be issued (by a call to subroutine CONTRL) to ship a part of the required type to the unit that reported the "hole." The discussion of subroutine CONTRL outlines the rules that are followed in different circumstances (see Section XI).

The last step is to place the vehicle number in the NORQ queue that contains the numbers of those vehicles assigned to that unit and missing that part. The vehicle number is ordered in the queue by the amount of time remaining until the vehicle would have been ready to go if the part had been available; the vehicle with the least time is first. For subroutine NORRPT to manage these queues, the array NORQ stores the vehicle number, the time remaining, and a pointer to the next report in the queue. The fourth element of the PARTS array (PARTS(PART, 4, BASE)) contains the pointer to the position in the NORQ array of the first vehicle that requires that type of part.

Whenever the "hole" has been filled, this subroutine is called through entry REDNOR to take the record out of the queue in NORQ. This is done after the tallies noted earlier are updated.

Subroutine REDPEO

This subroutine is used to reduce the number of maintenance and combat engineer personnel at a unit when some are shipped to another unit and to reorganize the number that remain after an attack. Subroutine SHPRES calls in the first instance and subroutine BOMB in the second. Calls to this subroutine prescribe the type (PEOP) and the number (NUM) of personnel to be withdrawn; NUM = 0 when the survivors of an attack are to be reallocated to the day and night shifts. Distinct procedures are used for maintenance personnel and for combat engineer personnel.

The first step in this subroutine is to identify whether personnel of the designated type are assigned to two or more unit subgroups, e.g., company teams or batteries. (The ALTPEO array provides the necessary data on the equivalent types of personnel.) If they are, the personnel are cross-levelled among the several subgroups in the proportions implied by the "target" force levels.[1] The next step is to establish what numbers will be on the day and night shifts after reorganization.

---

[1] "Target" force levels are generally taken to be the authorized strengths except that for each type of personnel the "target" can not exceed 99.

The new shifts are sized in the same proportions as the "target" force levels, except that no shift is allowed to be smaller than the "minimum shift size" entered with Card Type #21.

If some personnel at work during the present shift must be released, parts repairs are interrupted first; if more personnel are required, vehicle maintenance tasks are interrupted. If more people have been directed to be transferred than can be found, the number to be transferred is reduced accordingly; this situation can arise if personnel are being used in other than their "parent" shop (where they cannot be located readily).

The procedures for the combat engineer personnel are comparable except that they are all in a single organization and the choice of tasks to be interrupted is based on the facility priority list (Card Type #39); personnel are released from the lowest priority task first. When a combat engineering task is interrupted the work remaining to be done (i.e., the current damage level) is estimated on the assumption that the remaining work is the same fraction of the total job as the remaining time is of the total time. The quantities of unused building materials are estimated in the same manner and they are returned to stock.

Subroutine RESET

When the control variable EXTEND is initialized to unity, the simulation may be extended to an indefinite length and is not restricted to 65 days. This is done by resetting the various time values in the simulation data base at the end of each trial, but without reinitializing any of the resource status values; thus the second trial is just an extension of the first, and so forth. This subroutine performs all the necessary time adjustments when called by subroutine TRIALS.

Subroutine SHIFT

This subroutine is called at two hour intervals by subroutine MANAGE and changes the size of the on-duty work force for the maintenance personnel assigned to whichever work centers (shops) have a shift change at that time. Both the day and the night shifts are assumed to be 12 hours in length. Shifts that begin between midnight and 10:00 AM, inclusive, are designated the "day" shift. Using Card Type #19, the shift schedule is prescribed independently for each shop. The work schedules are the same on all units for shops of like number; however, the number of personnel on the different shifts is controlled independently for the different units using Card Type #21. Only vehicle maintenance personnel are treated in this subroutine; combat engineer personnel are assumed to pursue reconstruction tasks at a steady rate and are organized into shifts of equal size.

The basic function of this subroutine is to check whether more people are currently engaged than will be available on the next shift, and if so, to interrupt enough activities so that the required number of personnel may be released, or, if more people will be on duty, to attempt to assign the extra personnel to interrupted or waiting activities. The complications arise from personnel that may (1) be allowed to work a specified amount of overtime if they can complete their task within that time, or if they are engaged on a vehicle that has been scheduled for a late departure and (2) have been lent to another shop and will not be found when their parent shop is checked.

The first step taken when a shop changes shifts is to reset a flag and zero a counter in the sixth and seventh positions of the PEOPLE array. Then, for each shop, each parts repair job and each vehicle maintenance task is checked. At the first encounter with an as yet unchecked type of personnel, the flag is set to one and the net change in shift size is established. If the new shift is sufficient to handle all ongoing repairs and vehicle tasks, the flag is set to two, and the next activity is checked for any different personnel types that may be at work. If the follow-on shift cannot handle the current work load, parts repairs and on-vehicle tasks are interrupted (in that order) until a sufficient number are released, at which point their flag is set to

two. The most recently initiated parts repairs or on-vehicle tasks are interrupted first. The counter in the seventh position in PEOPLE maintains a record of the number of personnel that remain to be released.

If the personnel on a particular activity can finish their task within the allowed overtime period (for this decision it is presumed that the exact completion time is known), or if they are working on a vehicle that is scheduled for a late departure, they are allowed to continue; they are credited to the required reduction and subtracted from the "available" personnel for the subsequent shift. Thus at the beginning of a shift the number of personnel available can be a negative number equal to number of personnel that are working overtime; as each group is released, the "available" personnel remains at zero or less until fewer than the designated number on the next shift remain assigned.

When personnel have been lent to another shop that may have its shift change at a different time, the flag and the counter are still operative; when the various activities are checked and the "borrowed" personnel are noted, they will be released if their flag value is zero or one. Otherwise, the activity continues; in effect, members of the new shift take over for those on the previous shift.

To avoid overlooking personnel assigned to shops that have no activity under way at the time the shift is changed, maintenance personnel are next checked type by type, and the PEOPLE data are modified as appropriate, when their parent shop has a scheduled shift change and the personnel flag is still zero.

For shops that have had a net increase in work force (measured by the counter REM), subroutine CHECK is called to start any outstanding jobs.

The only exceptions to the preceding description occurs for the "ready line" shop--Shop #25--and the shops associated with the pre-mission tasks--reconfiguration, weapons loading, and refueling (Shops #27, 28, and 29, respectively). Personnel attached to those shops who must be released are required to complete their current task, without regard to allowable overtime. This is done because such tasks tend to be fairly short and because it seemed likely that such critical tasks would be completed in wartime.

Subroutine STRTSK

This subroutine manages the storage of unscheduled on-vehicle maintenance tasks in the RQDTSK and DEFTSK arrays. At the time a vehicle completes a mission and the unscheduled tasks are identified in subroutine PSTFLT, a tentative selection is made for the next mission and the tasks are separated into those that are required and those that may be deferred. Separate entry points are provided to store (STTASK) and to remove (REMTSK) a task, and a flag in the calling statement identifies the array to which the task belongs.

Each array maintains an ordered set of tasks for each vehicle; two pointers in the ACN array determine the positions in the RQDTSK and DEFTSK arrays where the first tasks are stored for each vehicle. The tasks are ordered as they are identified, and for the required tasks the sequential shop structure that is defined with Card Type #29 is preserved by entering the minus value of the first task identified for each group of shops whose work may be pursued simultaneously. The end of each set of tasks for a vehicle is identified by a zero entry in the task number position.

Function TTIME

This function selects the "true" time for a job on the basis of a mean task time and a time distribution that are specified in the calling statement. For both on- and off-vehicle maintenance tasks, the user is restricted to the use of nine distinct distributions; for intratheater transportation and communication delays, up to 15 distributions may be specified (i.e., six additional distributions).

Twenty-five data points are stored in the local DIST arrays to represent each distribution. Several log-normal and uniform distributions with different variance to mean ratios are available in FTIME in AURA and could be changed easily to satisfy special user requirements. These data are interpreted as 1000 times the ratio of the true value to the mean value. The entry selected is determined by the draw of a random number between 1 and 25. The true time value is returned in AURA time units (multiples of three minutes).

Provisions exist so that the user may add delays or speed-up factors to the true time calculation. The nominal task times generated in TTIME are modified by several control variables to represent such efforts to shorten and otherwise expedite jobs. If the mean time and the random variate are designated as TM and F, the actual task time is generated as:

$$\text{HURRY} \times \text{F}(\text{TM} - \text{REDUCE}) - \text{SAVE}$$

where the variables HURRY, REDUCE, and SAVE may be specified separately at each unit for on-vehicle tasks, premission tasks, parts repair jobs, munitions assembly jobs, and combat engineering tasks (see Card Type #17/2).

Subroutine WAIT

This subroutine manages the on- and off-vehicle jobs that must wait and are stored in the WAITSK array, in the same manner that subroutine INTRUP manages interrupted activities. Each shop at each unit has a pointer to the first and the last on-vehicle task and to the first and the last parts repair job that is waiting for action by that shop.

This subroutine is used only when the user wishes to have activities ordered in their queues by the value of the parameter TIME; to be so ordered, the control variable ORDWT is initialized to unity. The items are ordered such that the one with the lowest value of TIME is first. And as with the INTRUP subroutine, the value for the TIME variable need not be a time, per se; the specific definitions in use are explained in connection with the calling routines.

The mechanics of this subroutine are identical to those in INTRUP, except that the queues are maintained in the 7th, 8th, and 9th positions of the WAITSK array.

Additional Services

There are 11 additional services; five are used in conjunction with the INLIST subroutine for formatting the listing of input data (i.e., LIST1 through LIST5). Four of the services interpret AURA time to provide the time of day in AURA time units, or hours and minutes, and the day and the hour (TOD, HRMIN, DAY, and DATE). The last two functions control the time horizon for projecting vehicle supply and demand (function THF) and the length of the time intervals used in that process (function TU). The user may specify these factors with his entries on Card Type #4/2, or use the encoded default values. As currently coded, the default values for the time horizon are 8 hours between 4 AM and 4 PM, 12 hours from midnight to 4 AM; 16 hours from 8 PM till midnight, and 20 hours from 4 PM to 8 PM; the 16 time intervals are defined by the function TU.

XV. OUTPUT

In a simulation that involves multiple trials and as wide a variety of activities as AURA, a great abundance of data might be reported. The output options that are provided with AURA permit the user to examine a substantial portion of what we judged to be the more relevant results, but all possible outputs certainly are not available. For the additional, more specialized kinds of results that some users may find necessary for their particular problems, custom additions should be appended at the time they are required. Otherwise, the costs in time and dollars for storing the data, and the space for displaying them, would have to be borne by all users.

The current output options are controlled by the variables PRINT, STATFQ, CUMSTA, PPRINT, and SCROLL. Input information that precedes the simulation results in the printed output is discussed in Section XII. (The various debugging statements and outputs controlled by the variable TEST will not be discussed in this section.) For the individual trials, PRINT controls the data printed that relate both to the numbers of missions and maintenance tasks accomplished. STATFQ controls the collection and display frequency of shop performance statistics, including statistical data on the resource constraints that cause on-vehicle maintenance delays; these statistical data may be obtained separately for each trial, or the results may be aggregated over all trials, depending upon whether CUMSTA is 0 or 1. PPRINT controls the display of the numbers of serviceable and reparable spare parts at each unit, both at initialization and whenever shop statistics are printed.

SCROLL provides the user an opportunity to observe the behavior of individual vehicles in some detail. When SCROLL is used, a record of the daily activities for each of up to 24 consecutively numbered vehicles is listed at the end of each day for the number of days specified. The first vehicle is #1, unless otherwise specified. The four numbers listed immediately following (i.e., below) the vehicle number are the number of missions initiated that day, the number of the unit the vehicle is assigned to, a coded number summarizing the

vehicle's maintenance status, and the current number of "holes" in the vehicle. Following these data, the times for the beginning and end of each mission and for each on-vehicle task are listed, along with a description of the completed activities.

In addition to these various data that may be obtained for each trial, the final results also include a day-by-day record of the average number of missions accomplished, and the standard deviation thereof, for each mission and for each unit, when more than one trial is run.

1. OUTPUT CONTROLLED BY THE VARIABLE PRINT

The data provided for each trial for a particular value of the variable PRINT include all items down to and including those listed for that value on page 132. If a "1" appears in the tens column of the PRINT variable, the initial Type #50 Cards will be displayed; a "2" in the tens column of the PRINT variable means that all Type #50 Cards will be displayed.

2. OUTPUT CONTROLLED BY THE VARIABLE STATFQ

When STATFQ is initialized to a value greater than zero, data on the duration of vehicle maintenance tasks, parts repair jobs, support equipment repair jobs, and vehicle maintenance delays are stored using the subroutine TIMES. These data are printed at the end of each STATFQ days, at the end of each trial, and at the end of the simulation by calling subroutine DELAYS from subroutine OUTPUT. In each case, the results presented are based on the cumulative data to that point in the simulation if CUMSTA is 1; if CUMSTA is zero the results are cumulated independently for each trial. The results at the end of each trial also include the delay data for those activities that are still waiting at that time, on the assumption that all delays end at that time.

The first set of results presents the number of activities, and the average length and standard deviation of the time that they required, for on-vehicle tasks, for off-vehicle jobs, and for support equipment repair jobs at each shop at each unit.

OUTPUT DATA CONTROLLED BY THE VARIABLE PRINT

PRINT	OUTPUT DATA
-1	EOT: Storage array status if any overflows occur in one or more of the 18 dynamic storage arrays.
0	EOT: Cumulative platoons/tubes and vehicles committed to combat, demanded, and the ratio of vehicles committed to those demanded; totals for each unit and each mission.[a] EOT: Cumulative on-vehicle tasks, parts and support equipment repairs by unit and by shop. EOT: Reconstitution indices[b] and cumulative deadline (NMCS) hours at each unit.
1	EOT: Vehicles committed, demanded, and the percent of those demanded by unit and mission, ordered by priority. EOD: Vehicles possessed, lost, damaged, fillers, reserves, and transferred. EOD: Vehicles committed and damaged vehicles by unit. EOT: Daily reports listed for PRINT = 2.
2	EOD: Vehicles committed, demanded, and percent of those demanded by unit and mission. EOD: On-vehicle and off-vehicle tasks completed during the day by unit and by shop. EOD: Current supply of munitions by type. EOD: Numbers of tasks and repairs being processed, and the numbers of tasks waiting by unit and by shop. (also listed at noon if PRINT = 3) EOD: Status of ATE and dynamic storage, and spares shipments. EOT: Remaining supplies of munitions and spares. Number of deadlined (NMCS) vehicles by unit every six hours. Numbers of vehicles possessed, damaged, and those with one or more "holes" by unit, at three hour intervals.
3	EOD: Platoons/tubes committed and demanded by mission and unit. EOD: Number of vehicles committed to combat each hour by each unit. EOD: Numbers of repairs waiting, tasks and repairs interrupted. EOD: Cumulative manhours on vehicle tasks, parts, and support equipment repairs by shop and by unit. Current supply of spare parts at each unit every six hours. Notice of initiation of facility repairs.
4	The numbers of interrupted tasks and repairs at noon. Available munitions by type every six hours. Notice of completion of facility repairs.
5	Hourly listing of the number of vehicles waiting at each shop on each unit.
6	Numbers of personnel, support equipment, and parts for restricted types[c] of these resources are listed at noon for each unit.
7	Value of TEST is set to 14 every two hours when munitions assembly routines are accessed.
8	Same as for PRINT = 6 (i.e., does not include PRINT = 7 features) except that a record will be printed for all on-vehicle tasks still waiting at the EOT.[d]

EOT = End of trial
EOD = End of day

[a]Mission data are available by unit, vehicle type, mission type, and priority.

[b]The reconstitution indices provide a cumulative measure of how quickly vehicles were repaired and replenished for combat. The index is the average percentage of each unit's vehicles that were ready to fight within 2, 4, 6, and 8 hours after the previous mission.

[c]The data include PEOPLE(I,3,BASE) for I = 1-20 and 27-30; AGESTK(I,2,BASE) for I = 24; PARTS(I,J,BASE) for I - 1-24 and J = 1 and 2.

[d]Each such record begins with the notation "END WAIT" and is followed by the unit number, the resource class number and type number of the resource causing the wait, and the time in TTU that task has waited.

The standard time, or resource unconstrained time, as calculated during the input process in subroutine AVGTME, is also listed for the on-vehicle and off-vehicle activities; the values computed in AVGTME for the various vehicle types are weighted in the output by the numbers of missions achieved by the various vehicle types at each unit.

The second set of data provides a count of the ready vehicles that were canceled by crewmen shortage and a count of the additional numbers of crewmen that would have been needed to satisfy the minimum mission requirements.

The last set of data provides a statistical summary of the causes and the duration of vehicle maintenance delays. For each unit, for each of the other nine classes of resources, and for each individual resource type that caused an on-vehicle task to be delayed, the results include the number of such delays and the average value and standard deviation of their duration. If any of the vehicles have "holes" at the time of the report, the number of holes is listed with the parts data for each unit.

Data of the several types controlled by STATFQ are listed only when there are results to be reported; null data are suppressed.

RAND/N-1987-MRAL AURA USER'S MANUAL: VOL. I, PROGRAM FEATURES AND INTERACTIONS Shishko, Kamins